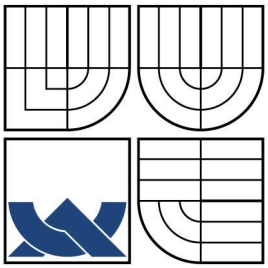
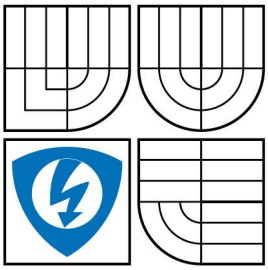


VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
BRNO UNIVERSITY OF TECHNOLOGY



FAKULTA ELEKTROTECHNIKY A KOMUNIKAČNÍCH TECHNOLOGIÍ
ÚSTAV TELEKOMUNIKACÍ

FACULTY OF ELECTRICAL ENGINEERING AND COMMUNICATION
DEPARTMENT OF TELECOMMUNICATIONS

IMPLEMENTACE KVALITY SLUŽBY DO KONVERGOVANÉ SÍTĚ

IMPLEMENTATION OF QUALITY OF SERVICE IN A CONVERGED NETWORK

BAKALÁŘSKÁ PRÁCE
BACHELOR´S THESIS

AUTOR PRÁCE
AUTHOR
DANIEL ZVOLENSKÝ

VEDOUCÍ PRÁCE
SUPERVISOR
DOC.ING. VLADISLAV ŠKORPIL, CSC

BRNO 2008

LICENČNÍ SMLOUVA
POSKYTOVANÁ K VÝKONU PRÁVA UŽÍT ŠKOLNÍ DÍLO

uzavřená mezi smluvními stranami:

1. Pan/paní

Jméno a příjmení: Daniel Zvolenský

Bytem: Vajanského 7, 91701, Trnava

Narozen/a (datum a místo): 1.5.1986

(dále jen „autor“)

a

2. Vysoké učení technické v Brně

Fakulta elektrotechniky a komunikačních technologií

se sídlem Údolní 244/53, 602 00, Brno

jejímž jménem jedná na základě písemného pověření děkanem fakulty:

.....

(dále jen „nabyvatel“)

Čl. 1
Specifikace školního díla

1. Předmětem této smlouvy je vysokoškolská kvalifikační práce (VŠKP):
   - □ disertační práce
   - □ diplomová práce
   - □ bakalářská práce
   - □ jiná práce, jejíž druh je specifikován jako .....

(dále jen VŠKP nebo dílo)

Název VŠKP:	Implementace kvality služby do konvergované sítě
Vedoucí/ školitel VŠKP:	doc. Ing. Vladislav Škorpil, Csc.
Ústav:	Ústav telekomunikací
Datum obhajoby VŠKP:	

VŠKP odevzdal autor nabyvateli v*:

□ tištěné formě – počet exemplářů 1

□ elektronické formě – počet exemplářů 1

* hodící se zaškrtněte

2. Autor prohlašuje, že vytvořil samostatnou vlastní tvůrčí činností dílo shora popsané a specifikované. Autor dále prohlašuje, že při zpracovávání díla se sám nedostal do rozporu s autorským zákonem a předpisy souvisejícími a že je dílo dílem původním.
3. Dílo je chráněno jako dílo dle autorského zákona v platném znění.
4. Autor potvrzuje, že listinná a elektronická verze díla je identická.

Článek 2
Udělení licenčního oprávnění

1. Autor touto smlouvou poskytuje nabyvateli oprávnění (licenci) k výkonu práva uvedené dílo nevýdělečně užít, archivovat a zpřístupnit ke studijním, výukovým a výzkumným účelům včetně pořizovaní výpisů, opisů a rozmnoženin.
2. Licence je poskytována celosvětově, pro celou dobu trvání autorských a majetkových práv k dílu.
3. Autor souhlasí se zveřejněním díla v databázi přístupné v mezinárodní síti
   - □ ihned po uzavření této smlouvy
   - □ 1 rok po uzavření této smlouvy
   - □ 3 roky po uzavření této smlouvy
   - □ 5 let po uzavření této smlouvy
   - □ 10 let po uzavření této smlouvy

   (z důvodu utajení v něm obsažených informací)
4. Nevýdělečné zveřejňování díla nabyvatelem v souladu s ustanovením § 47b zákona č. 111/ 1998 Sb., v platném znění, nevyžaduje licenci a nabyvatel je k němu povinen a oprávněn ze zákona.

Článek 3
Závěrečná ustanovení

1. Smlouva je sepsána ve třech vyhotoveních s platností originálu, přičemž po jednom vyhotovení obdrží autor a nabyvatel, další vyhotovení je vloženo do VŠKP.
2. Vztahy mezi smluvními stranami vzniklé a neupravené touto smlouvou se řídí autorským zákonem, občanským zákoníkem, vysokoškolským zákonem, zákonem o archivnictví, v platném znění a popř. dalšími právními předpisy.
3. Licenční smlouva byla uzavřena na základě svobodné a pravé vůle smluvních stran, s plným porozuměním jejímu textu i důsledkům, nikoliv v tísni a za nápadně nevýhodných podmínek.
4. Licenční smlouva nabývá platnosti a účinnosti dnem jejího podpisu oběma smluvními stranami.

V Brně dne: …….

…….. ……
Nabyvatel Autor

Anotácia

Cieľom tejto práce je v prostredí Network Simulator verzie 2 vytvorenie IP siete s vhodnou topológiou, v ktorej sa bude simulovať paketový hlasový tok medzi účastníkami, v úvážení použitia kompresie podľa doporučenia G.723. A taktiež vytvorenie simulácie prenosu videa v reálnom čase s uvážením doporučenia H.263.

Teoretická časť tejto práce je venovaná konvergovaným sieťam a implementácii kvality služieb (QoS) do konvergovanej siete.

Praktická časť obsahuje zoznámenie sa s programom Network Simulator verzie 2, ktorý umožňuje simuláciu chovania sa siete.

Klúčové slová: Network Simulator verzie 2, G.723, H.263, RTP/RTCP, IPv6, QoS, VoIP, UDP

Anotace

Cílem této práce je v prostředí Network Simulator verze 2 vytvoření IP sítě s vhodnou topológií, v ktoré se bude simulovat paketový hlasový tok mezi účastníkmi, v úvážení dle doporučení G.723. A taky vytvoření simulace přenosu videa v reálnem čase s uvážením doporučení H.263.

Teoretická část této práce je věnována konvergovaným sítím a implementáci kvality služeb (QoS) do konvergované sítě.

Praktická část obsahuje seznámení se s programem Network Simulator verze 2, který umožňuje simulace chování se sítě.

Klíčová slova: Network Simulator verze 2, G.723, H.263, RTP/RTCP, IPv6, QoS, VoIP, UDP

Abstrakt

The point of this work is the creation of IP network with a proper topology, in Network Simulator version 2, where voice packet stream with recommended compresion by G.723 standard and real-time video by H.263 standard will be simulated.

Theoretical part of this work consists of convergence of nets with an implementation of Quality of Services description.

In practical part of this work is identification with bacis commands in program Network Simulator 2, which make possible simulation of network behaviour.

Keywords: Network Simulator version 2, G.723, H.263, RTP/RTCP, IPv6, QoS, VoIP, UDP

Prehlásenie

Prehlasujem, že svoju bakalársku prácu na tému „Implementace kvality služby do konvergované sítě“ som vypracoval samostatne pod vedením vedúceho bakalárskej práce a s použitím odbornej literatúry a ďalších informačných zdrojov, ktoré sú všetky citované v práci a uvedené v zozname literatúry na konci práce.

Ako autor uvedenej bakalárskej práce ďalej prehlasujem, že v súvislosti s vytvorením tejto bakalárskej práce som neporušil autorské práva tretích osôb, obzvlášť som nezasiahol nedovoleným spôsobom do cudzích autorských práv osobnostných a som si plne vedomý následku porušenia ustanovenia § 11 a nasledujúceho autorského zákona č. 121/2000 Sb., vrátane možných tresnoprávnych dôsledkov vyplývajúcich z ustanovenia § 152 trestného zákona č. 140/1961 Sb.

V Brně dňa

(podpis autora)

Poďakovanie

Ďakujem vedúcemu bakalárskej práce Doc. Ing. Vladislavu Škorpilovy, Csc., za veľmi užitočnú metodickú pomoc a cenné rady pri spracovávaní tejto bakalárskej práce.

V Brne dňa

(podpis autora)

Abecedný zoznam skratiek

ACELP	Algebraic Code Excited Linear Prediction
ADPCM	Adaptive Differential Pulse Code Modulation
ATM	Asynchronous Transfer Mode
CAC	Call Admission Control
CRC	Cyclic Redudancy Check
CSMA/CD	Carrier Sense Multiple Access with Collision Detection
DARPA	Defence Advanced Research Projects Agency
DAD	Duplicate Address Detection
DHCP	Dynamic Host Configuration Protocol
DSCP	Differential Service Code Point
FCS	Frame Check Sequence
FR	Frame Relay
IEEE	Institute of Electrical and Electronics Engineering
IP	Internet Protocol
IPv4	Internet Protocol version 4
IPv6	Internet Protocol version 6
ISDN	Integrated Service Digital Network
ISO	International Standards Organization
ITU-T	International Telecommunication Union
LAN	Local Area Network
MGCP	Media Gateway Control Protocol
MPLS	Multi Protocol Label Switching
NGN	Next Generation Network
OSI	Open System Interconnection
PAD	Peripheral Arterial Disease
PCM	Pulse Code Modulation
PDU	Protokol Data Unit
PNNI	Private Network – Network Interface
RAS	Remote Access Services
RIP	Routing Information Protocol
RSVP	ReSerVation Protocol
RTP	Real-time Transsport Protocol
RTCP	Real-time Transport Control Protocol
QoS	Quality of Services
SFD	Start of Frame Delimeter
SIP	Session Initiation Protocol
SNMP	Simple Network Management Protocol
TCP	Transmission Control Protokol
TDM	Time Division Multiplex
TFTP	Trivial File Transfer Protocol
ToS	Terms of Services
UDP	User Datagram Protocol
VLAN	Virtual Local Area Network
VoIP	Voice over Internet Protocol
VoATM	Voice over ATM

Obsah str.

Zoznam obrázkov:

Zoznam tabuliek:

Úvod

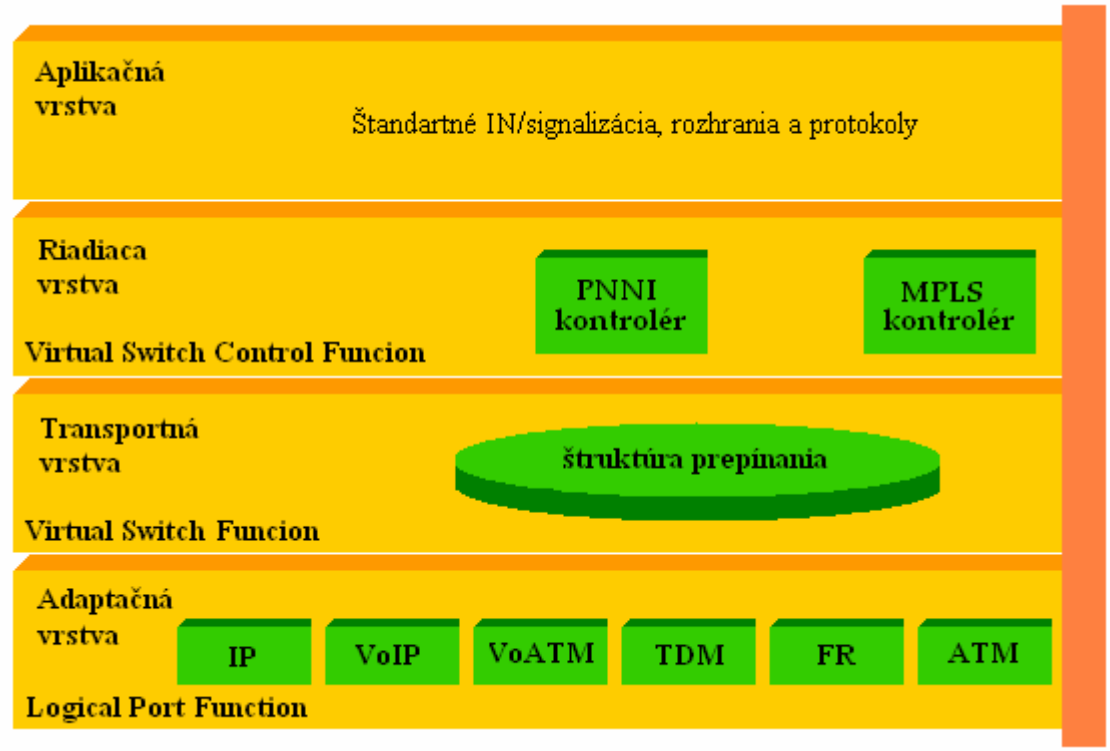
Cieľom tejto práce je zoznámenie sa s implementáciou kvality služieb (QoS) do konvergovanej siete, tj. siete prenášajúcej spoločne hlasovú a dátovú službu, a simulácia vybraných partii tejto problematiky. Simulácia je koncipovaná ako laboratórna úloha v prostredí Network Simulator 2.

Prvá časť projektu obsahuje popis a skladbu používanej dátovej jednotky a informácie o protokoloch, ktoré sa na prenose multimédii podieľajú a dôvody, prečo sa používajú. Taktiež sa zaoberá problematikou kvality služieb a jej architektúrou.

Druhá časť obsahuje zoznámenie sa s programom Network Simulator verzie 2, ktorý umožňuje simuláciu chovania sa siete. Vytvorenie simulácie prenosu hlasu a videa cez konvergovanú sieť, analyzovanie oneskorenia, jitteru a stratovosti paketov, simulácia poruchy linky.

1 Konvergovaná sieť [6]

Termín „konvergencia“ je tradične chápaný ako proces postupného splývania dvoch doposiaľ oddelených prenosových infraštruktúr – telekomunikačných a dátových sietí, motivovaný najmä technologickými dôvodmi.
Konvergované siete – označované ako NGN (siete budúcej generácie) – zjednocujú oba typy sietí (telekomunikačnú a dátovú) do jednej všeobsiahlej siete s plne centralizovaným riadením, založenej na smerovaní a prepájaní paktov.
Architektúra a koncepcia NGN siete podľa firmy CISCO® je zobrazená na obr.1.



Obr.1.: Architektúra NGN podľa firmy CISCO®

1.1 Súpis skratiek

PNNI	Private Network – Network Interface verejná telefónna sieť s prepájaním okruhov	VoIP	Voice over IP prenos hlasu cez IP
MPLS	Multi Protocol Label Switching viacprotokolové prepájanie návestím	VoATM	Voice over ATM prenos hlasu cez ATM
IP	Internet Protocol časové združovanie kanálov	TDM	Time Division Multiplex
ATM	Asymetric Digital Subscribe Line nesúmerná digitálna účastnícka linka	FR	Frame Realy prenos rámcov

Konvergovaná sieť teda umožňuje prenos hlasu a videa po dátovej sieti s ostatnými dátami cez internetový protokol IP. Výhodou je, že prevádzkovateľ využíva len jednu spoločnú sieť, čím sa mu zmenšujú požiadavky na náklady. Aby prenos hlasu a obrazu bol kvalitný, vytvorili sa súbory protokolov, ktoré túto skutočnosť zaisťujú. Ich pozornosť im bude venovaná v nasledujúcich kapitolách [7].

2 ETHERNET [1]

2.1 Napájanie po Ethernete

Brať elektrický prúd pre desiatky až stovky zariadení pre IP komunikáciu, ako sú IP telefóny, zo zásuviek na stene nie je praktické riešenie. Štandard 802.3af dnes umožňuje prepínačovej infraštruktúre LAN napájať takéto zariadenia cez Ethernetový kábel.
K dispozícii sú i riešenia umožňujúce zároveň napájanie inteligentne riadiť a tak šetriť elektrickú energiu.

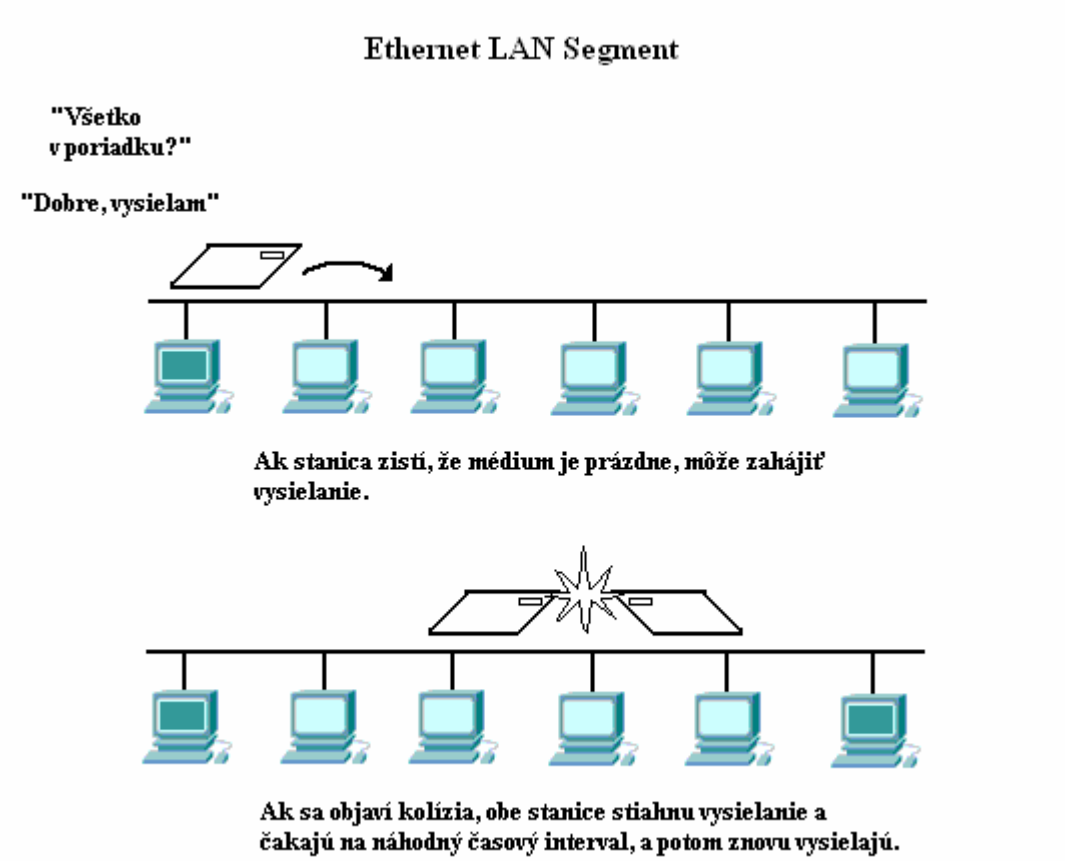
Vzhľadom k tomu, že špecifikácia štandardu 802.3af dovoľuje šíriť elektrickú energiu v sieti len s niekoľkými pevne danými príkonovými stupňami bez ohľadu na skutočnú náročnosť jednotlivých zariadení, vznikli technológie, ktoré umožňujú užívateľom upraviť príkon podľa faktických potrieb aktívnych prvkov siete. Vďaka presnejšiemu prideľovaniu príkonu sa taktiež optimalizuje hustota portov, čo vedie ku zníženiu počtu prepínačov a zdrojov elektrickej energie.

2.2 Ethernet

Prvá LAN na svete bola originálna verzia Ethernetu. Robert Metcalfe a jeho spolupracovníci z Xerox ho vytvorili viac ako tridsať rokov dozadu. Za nasledujúce desaťročie, Xerox spolu so spoločnosťami Intel a Digital Equipment vydali Ethernet verzie dva (rok 1982). Od toho obdobia sa Ethernet stal štandardnou dominantou sieťových technológii. Stalo sa tak najmä vďaka ekonomickému pomeru, ktorý zabezpečil, že priemerná cena za Ethernetový port je dnes nižšia ako cena Token Ring portu. V skutku, Ethernet sa stal viac de facto štandardom, pretože veľa dnešných zariadení má integrovaný práve Ethernetový sieťový adaptér na základných doskách.

2.3 Ethernet architektúra

Ethernet bol vytvorený k zriadeniu siete s veľkým počtom počítačov, ktoré boli pripojené na zdielanej „bus“ topológii.
Prvá verzia Ethernetu pripájala prístup k médiu metódou známou ako CSMA/CD (Carrier Sense Multiple Access with Collision Detection). Zariadenia zdieľajúce Ethernet LAN segment odpočúvajú prevádzku na linke a čakajú na vysielanie dát pokým nie je médium voľné. Ak dve stanice vysielajú dáta v rovnaký časový interval, ich pakety sa zrazia a oba prenosy sú prerušené. Stanice musia odstúpiť od vysielania a vyčkať na náhodnú časovú periódu, ktorá im určí ich vysielacie poradie. CSMA/CD je algoritmus pre odpočúvanie prevádzky na médiu s detekciou kolízii a s prerušením vysielania. CSMA/CD algoritmus sa teda správa ako správca, ktorý určuje prístup k médiu za účelom zabezpečiť úplnosť vysielania. Obr.2 ilustruje CSMA/CD proces.



Obr. 2.: Ethernet prístup kontrolovaný dostupnosťou média a detekciou kolízii

Pretože médium je zdielané, každé zariadenie na Ethernet LAN segmente obdrží správu a skontroluje či sa cieľová adresa zhoduje s jej vlastnou adresou. Ak sa zhodujú, správa je prijatá a nastane proces cez sedem vrstiev ISO/OSI. Tým sa uskutočnilo sieťové pripojenie. Ak adresa nesúhlasí, paket je zahodený.

2.4 Ethernet implementácia

Ethernet je neodmysliteľne najmenej nákladný vďaka jeho chaotickému charakteru architektúry. Inak povedané, elektroniku bežiacu na Ethernete je ľahšie zhotoviť, pretože Ethernet nevyžaduje kontrolu všetkého (stará sa iba o kolízie).

Väčšina prevádzky v Internete je vytvorená a ukončená práve Ethernetovým spojením. Od počiatku sedemdesiatich rokov ako bol Ethernet vyvinutý, sa musel vysporiadať s požiadavkou pre vysokorýchlostné LAN. Keď optické vlákna boli predstavené, Ethernet prispôsobil túto novú technológiu pre zlepšenie šírky pásma a zmenšenie počtu chýb, ktoré optika ponúka. Preto, dnes na rovnakom protokole, ktorý kedysi prenášal dáta za 3 Mbps môžeme prenášať dáta za 10 Gbps.

Ethernet má niekoľko zavedených štandardov. Prvotný Ethernet bežal na 3 Mbps cez koaxiálny kábel alebo cez 10Base-T krútenú dvojlinku (*T* – twist). Fast Ethernet beží na 100 Mbps cez 100BaseTX alebo 100baseFX optickom vlákne (*F* - fiber). Gigabit Ethernet beží na 1 Gbps (alebo 10 Gbps) cez 1000BaseTX alebo 1000BaseFX kabeláži. Nasledujúca tabuľka 1 zobrazuje všetky štandardné typy Ethernetu.

Tab. 1.: Typy Ethernetu

Ethernet typ	Šírka pásma	Typ káblu	Duplex	Maximálna dĺžka kabeláže
10Base-5	10 Mbps	tlstý koaxiál	half	500 m
10Base-2	10 Mbps	tenký koaxiál	half	185 m
10Base-T	10 Mbps	kat3/kat5 UTP	half	100 m
100Base-TX	100 Mbps	kat5 UTP	half	100 m
100Base-TX	200 Mbps	kat5 UTP	full	100 m
100Base-FX	100 Mbps	mnohovidové vlákno	half	400 m
100Base-TX	200 Mbps	mnohovidové vlákno	full	2 km
1000Base-T	1 Gbps	kat5 UTP	full	100 m
1000Base-TX	1 Gbps	kat6 UTP	full	100 m
1000Base-SX	1 Gbps	mnohovidové vlákno	full	550 m
1000Base-LX	1 Gbps	jednovidové vlákno	full	2 km
10Gbase-CX4	10 Gbps	twin-axial	full	100 m
10Gbase-T	10 Gbps	kat6/kat7 UTP	full	100 m
10Gbase-LX4	10 Gbps	mnohovidové vlákno	full	300 m
10Gbase-LX4	10 Gbps	jednovidové vlákno	full	10 km

V 10Base-T sieťach je typickým Ethernet zariadením HUB. Ten umožňuje zdieľanie média. Pretože médium je zdieľané, iba jedna stanica môže úspešne vysielať v danom čase. Tento spôsob spojenia je známy ako *half-duplex* komunikácia. Využíva prístupovú metódu k médiu CSMA/CD, ktorá je popísaná vyššie.

Významným zdokonalením, ktoré rozšírilo LAN realizáciu bolo zavedenie switchov namiesto HUBov v Ethernet sieťach. Tento vývoj je úzko spätý s vyvinutím 100Base-TX Ethernet. Switche môžu kontrolovať tok dát rozdelením na jednotlivé porty a odosielanie frejmov iba do danej cieľovej stanice.
Switche redukujú počet zariadení prijímajúcich každý frejm, čím minimalizujú možnosť kolízie. Tento spôsob spojenia je známy ako *full-duplex* komunikácia.

2.5 Kódovanie pre jednotlivé typy Ethernetu [8]

Spomínaný štandartd10Base-T využíva **kódovanie Manchester** cez dva páry netienenej dvojlinky (tzv. twisted-pair). *Manchester* je kódovanie s návratom k nule.

Ethernet so štandardom 100Base-T, ktorý podporuje prenášanie dát cez UTP kábel kategórie 5, využíva **4B/5B kódovanie**. Táto metóda je navrhnutá s limitovaným počtom po sebe idúcich núl. Začína od ľavého najvyššieho bitu, každý 4. bit je nahradený piatym. *4B/5B*

kódovanie dát je potom prenášané cez sieť využívajúc NRZI prenosový kód pre každý poslaný bit.

1000Base-T Ethernet používa **4D-PAM5 kódovanie** k dosiahnutiu kapacity 1 Gbps dát. Toto kódovanie umožňuje prenášanie signálov cez štyri linky súčastne. A to vďaka prekladaniu ôsmych bitov dát súčastne prenesených do štyroch kódových symbolov (4D), ktoré sú odoslané cez médium, jeden každým párom, ako 5-level Pulzne Amplitúdovo Modulované (PAM5) signály. To znamená, že každý symbol odpovedá dvom bitom dát. Pretože informácia putuje súčastne cez štyri linky, systém delí frejmy odoslané a znovu zostavené ich príjemcom.

2.6 Frejm - zapúzdrenie paketu [1]

Protokolová dátová jednotka (PDU) sieťovej podvrstvy ISO/OSI je zapúzdrená do Ethertnetového frejmu, ktorý sa skladá z hlavičky a traileru.

Ethernetová hlavička a trailer obsahujú niektoré časti informácii, ktoré sú použité Ethernetovým protokolom. Každú časť frejmu voláme pole. Sú známe dva štýly Ethernetových frejmov**: IEEE 802.3** (originál) a upravený **IEEE 802.3** (Ethernet).

Rozdiely medzi týmito štýlmi sú minimálne. No ten podstatný rozdiel je pridanie SFD (Štart Frejmového Oddeľovača) a malé zmeny v dĺžke poľa ako je zobrazené na obr. 3 [2].

Obr. 3.: Porovnanie rozdielov Ethernetových štruktúr- 802.3 a Ethernet

2.6.1 Veľkosť Ethernetového frejmu [1] , [2]

Originálny štandart Ethernetu definuje minimálnu veľkosť frejmu ako 64 bajtov a maximálnu veľkosť frejmu ako 1518 bajtov. Ten obsahuje všetky byty od poľa cieľovej MAC adresy až po pole pre overovaciu frejmovú sekvenciu (FCS). Polia preambula a štart frejmového oddelovača (SFD) nie sú zahrnuté, keď hovoríme o veľkosti frejmu. Štandard IEE 802.3ac, vydaný v roku 1998, zväčšil maximum dostupnej veľkosti frejmu na 1522 bajtov. Stalo sa tak kvôli prispôsobeniu sa novej technológii nazvanej Virtual Local Area Network (VLAN).

Ak je veľkosť prenášaného frejmu menšia ako minimum, alebo väčšia ako maximum, prijímané zariadenie frejm zahodí. Zahodené frejmy sú ako dôsledok kolízie alebo nežiaducich signálov, a preto sú považované za neplatné.

2.6.2 Jednotlivé časti Ethernetového frejmu

- **Preambula a Štart Frejmového Oddelovača**

Preambula a SFD polia sú použité na synchronizáciu medzi vysielacím a prijímaným zariadením. Týchto prvých osem bajtov frejmu je použitých k autorizácii prijímaných uzlov. V podstate, hovoria príjemcovi, že môže prijať nový frejm.

- **Cieľová MAC Adresa**

Táto adresa slúži ako identifikátor pre určeného príjemcu. Ak sa MAC adresa príjemcu zhoduje s adresou vo frejme, zariadenie frejm príjme.

- **Zdrojová MAC Adresa**

Táto adresa identifikuje odosielateľa frejmu. Switche taktiež používajú túto adresu k vyhľadávaniu v ich tabuľkách.

- **Dĺžka/Typ**

Definujú presnú dĺžku dátového poľa frejmu. Tento parameter je využitý neskôr ku kontrole správneho prijatia správy. Pole Typ popisuje aký typ protokolu je implementovaný.

- **Dátové Pole**

Obsahuje zapúzdrené dáta od vyšších vrstiev ISO/OSI, ktoré boli generované do paketu. Paket je teda obalený frejmovou hlavičkou a trailerom. Všetky frejmy musia byť aspoň 46 bajtov dlhé. Ak je zapúzdrený paket menší, je použitý tzv. PAD, ktorý zaplní prázdne miesto a tým sa získa minimálna veľkosť frejmu .

- **Overovacia Sekvenica Frejmu**

Toto pole je použité k detekovaniu chýb vo frejme. Využíva tzv. periodicky nadbytočné overovanie (CRC). Ak zariadenie príjme frejm, generuje CRC pre overenie chýb. Ak je spozorované, že dáta boli pozmenené, frejm je zahodený. Zmena dát môže byť spôsobená chybou v elektrických signáloch, ktoré prezentujú jednotlivé bity.

3 IPv6 [1]

Pokiaľ adresovanie v Internete sa zdá byť bez problémov, šokujúcou skutočnosťou je, že sa nachádzame mimo adresný priestor. Naozaj, aj keď to môže znieť prekvapujúco a IPv4 s 32 bitovým adresným priestorom ponúka viac ako dosť adries (v skutočnosti ich je okolo 4 miliárd), potrebujeme väčší adresný priestor.

Riešenie prináša IPv6, taktiež známa ako IPng (ng znamená novej generácie). IPv6 ponúka až 128 bitový priestor čo predstavuje 340 282 366 920 938 463 463 374 607 431 768 211 456 možných adries.

3.1 Formát

Ako bolo spomenuté skôr, IPv6 používa 128 bitový adresný priestor. Adresy sú vyjadrené ôsmymi poliami po 16 bitov v hexadecimálnom formáte (0000-FFFF).

x : x : x : x : x : x : x : x

Príklad tohoto formátu môže byť

FESC : BA98 : 7654 : 3210 : FEDC : BA98 : 7654 : 3210

alebo

1080 : 0 : 0 : 0 : 8 : 800: 200C : 417A.

IPv6 adresy môžu byť vyjadrené 3 spôsobmi:

- Najviac používanou metódou je jednoduché vloženie hodnôt do každého z ôsmich políčok:

1070 : 200 : 0 : 0 : 900 : 300C : 618A.

Ako vidno na príklade, nie je potrebné použiť „0“ pred „200“. Teda môžme napísať namiesto „0200“ zjednodušene iba „200“.

- V niektorých prípadoch IPv6 adresy budú obsahovať dlhý rad núl. Preto môžeme použiť namiesto 0000:0000:0000:0000:0000:0000:0000:1 zápis 0:0:0:0:0:0:0:1. Niekedy je taktiež výhodnejšie použiť pre skupinu núl nasledujúci zápis „**::**“. IPv6 vie rozlíšiť, že má nasledujúci znak „**::**“ nahradiť nulami.

Nahradenie znaku „**::**“ zobrazuje nasledujúca tabuľka:

Tab. 2.: Zápis znaku „**::**“ v IPv6

1070:200:0:0:900:300C:618A	1070:200::900:300C:618A
FF01:0:0:0:0:0:0:100	FF01::100
0:0:0:0:0:0:0:1	::1
0:0:0:0:0:0:0:0	::

- Poslednou metódou je takzvaný prechod, pokým sa adaptuje IPv6. To znamená, že je použitý formát, ktorý je zmiešaný z IPv6/IPv4. Tento formát kombinuje obe verzie IP a je zobrazený nasledovne:

x : x : x : x : x : x : x : d.d.d.d.

V tomto príklade je „x“ hodnota v hexadecimálnom tvare a „d“ je hodnota v decimálnom tvare. Napr.:

0 : 0 : 0 : 0 : 0 : FFFF: 129.144.40.20

alebo môžeme nuly nahradiť znakom „**::**“ ako je spomenuté vyššie

:: FFFF : 129.144.10.20

3.2 IP prefix

Prefixová vzdialenosť je číslo bitov v adrese, ktoré nám ponúkajú časť siete. IPv6 prefix je teda časť adresy, ktorá reprezentuje koľko bitov (z ľavej strany) reprezentuje meno siete. Ostatné bity reprezentujú hostov. IPv6 prefix môže mať nasledujúci zápis:

1080 : 0 : 0 : 0 **: 8 : 800: 200C : 417A /64**

kde **/64** je náš prefix a znamená, že 64 bitov je použitých pre sieť (farebne označená časť) a zvyšných 64 bitov je možné využiť pre hostov.

3.3 Typy adresovania

Poznáme tri možné typy adresovania v IPv6:

- **Unicast**
  - **Globálne adresy**
  - **Lokálne adresy**
  - **IPv4 mapované Ipv6 adresy**
  - **IPv4 kompatibilné s IPv6 adresy**
- **Anycast**
- **Multicast**

3.4 Konfigurácia

IPv6 využíva vlastnosť nazvanú *prostá autokonfigurácia.* Je to obdobné ako u DHCP kde sú IP adresy automaticky prideľované, no rozdiel je v tom, že nie je použitý server ani špeciálna aplikácia. Použitím DHCP sa každý router, používajúci IPv6 adresovanie, stane „správcom“ IP adries v sieti, ku ktorej je pripojený. K predchádzaniu duplikácie adries IPv6 používa vlastnosť nazvanú *detekcia duplikovaných adries* (DAD).

4 VoIP [5] , [3]

Voice over Internet Protocol (VoIP) predstavuje alternatívnu hlasovú službu na prenos hovoreného slova prostredníctvom dátových sietí konvertovaním a kompresiou hlasu na dátové pakety, pričom na konci dátovej cesty sa spätným procesom z paketov vyrobí pôvodná hlasová stopa.

Prenos hlasu po IP vyžaduje, aby sa IP sieť chovala podobne (ale iba pre prenos paketov) ako tradičná telefónna sieť. Preto je treba signalizácia, ktorá umožňuje požadovanú kvalitu komunikácie medzi koncovými terminálmi a sieťou a overuje potrebu bezpečného spojenia

s garantovanou úrovňou kvality služieb *QoS*. Kapacita siete nie je všetko, čo hlasové služby môžu potrebovať. Prenos v reálnom čase a interaktívne aplikácie obecne sú citlivé na oneskorenie a kolísanie oneskorenia v sieťach. Preto pre tieto aplikácie nestačí mať iba dostatočnú šírku pásma, pretože to taktiež môže spôsobiť zahltenie bežného dátového prenosu a prenosu v reálnom čase a príjemca tak pociťuje komunikáciu ako nie veľmi kvalitnú.

Požiadavky na sieť pre hlasové služby teda jednoznačne vyplývajú zo skúseností s klasickou telefónnou sieťou:

- trvalá dostupnosť služby
- jednosmerné oneskorenie pod 150 ms
- privátnosť hovorov (ochrana proti odpočúvaniu)
- garantovaná vysoká kvalita a odozva v reálnom čase

Hlas je kritická aplikácia, ktorá vyžaduje maximálnu dostupnosť a spoľahlivosť siete (dostupnosť modernej telefónnej siete sa udáva v 99,999%). Pre porovnanie – len malá časť dátového prenosu je podobne kritická [3].

4.1 Hlasový prenos

Hlasový prenos sa od dátového prenosu značne nelíši svojim charakterom a požiadavkami. Na rozdiel od dát, ktoré sú zhlukové, potrebuje hlas menšiu, ale zato konštantnejšiu šírku pásma (12-106 kbps v závislosti na rýchlosti vzorkovania, kodeku a réžii na linkovej vrstve). Citlivosť na oneskorenie je u hlasu výrazne vyššie než u dát. Maximálna jednosmerná latencia by sa mala pohybovať pod 180 ms a maximálne kolísanie oneskorenia pod 30 ms. Citlivosť na stratu paketov je pri hlase menšia než u dát. Maximálna doporučená hodnota sa pohybuje okolo 1%.

4.2 Kodeky hlasového signálu

Analógový hlasový signál je pre prenos v dátových sieťach najprv *digitalizovaný*. K tomu sa využíva *vzorkovanie* kmitočtov 8 kHz a následné *kvantovanie*, pri ktorom sa každý navzorkovaný prvok priradí danej kvantizačnej úrovni. Potom nasleduje *kódovanie*, tj. zariadnie pre kódovanie/dekódovanie a súčastne kompriesiu/dekompresiu hlasového signálu, ktoré sa uskutočňuje podľa nasledujúcich typov kodekov:

- **G.711** – základné kódovanie PCM, výstupný tok s rýchlosťou 64kbps, rámec 0,125 ms, oneskorenie 0,75 ms; základný kodek pre všetky zariadenia pre VoIP (kompresia z 12 na 8 bitov)

- **G.723.1** – kódovanie ACELP, prenosová rýchlosť 5,3 kbps, veľkosť vzorku 30 ms, oneskorenie 30 ms

- **G.729** – kódovanie CS-ACELP, výstupný tok s rýchlosťou 8 kbps, rámec 10 ms, oneskorenie 10 ms; najpopulárniejší kodek z hľadiska pomeru kvality, oneskorenia a výkonu

4.2.1 Kodek G.723

G.723 je širokopásmový hovorový kodek štandardu ITU-T a je rozšírenou verziou doporučenia *G.721*, zahrňujúceho diferenciálnu pulznú kódovú moduláciu na 24 a 40 kbps pre digitálny okruh mnohonásobných aplikácii.

4.2.2 Kodek G.723.1

G.723.1 je audio kodek pre hlas, ktorý komprimuje audio hlas do 30 ms frejmov. Najviac používaný je vo VoIP aplikaciách, pretože má malé požiadavky na šírku pásma. Hudba alebo tóny ako DTMF (Dual-Tone Multi-Frequency) alebo fax nemôžu byť spoľahlivo prenášané cez tento kodek, a preto sa využívajú iné metódy ako *G.711* alebo *out-of-band* [3].

4.3 Transportné protokoly pre VoIP

Pre prenos v reálnom čase, medzi ktoré patrí aj IP telefónia, je nevyhnutné zaistiť rovnomernosť toku paketov, čo znamená, že časové oneskorenie jednotlivých paketov nemá kolísať – musí sa zachovať vzájomný vzťah medzi paketmi v rámci danej relácie. Ak sa časové odstupy medzi jednotlivými prijímanými paketmi výrazne menia, môže mať aplikácia v reálnom čase nekvalitný výstup, alebo dokonca môže dôjsť k jej kolapsu.

Spoľahlivý transportný protokol *TCP* nie je vhodný pre prenos paketov v reálnom čase, pretože neposkytuje časové údaje o odosielaní jednotlivých paketov a ani nepodporuje skupinové vysielanie, ktoré sa u aplikácii v reálnom čase veľmi často používa. Najnevýhodnejšou vlastnosťou *TCP* pre *VoIP* je jeho zabudované riadenie chýb – automaticky opätovný prenos pri strate alebo doručení chybného paketu. Pri prenose v reálnom čase je naopak potreba stratený alebo poškodený paket ignorovať, pretože jeho opätovný prenos je zbytočný a paket je už nepoužiteľný.

UDP je vhodnejší transportný mechanizmus pre prenos v reálnom čase, ale pretože nemá niektoré požadované vlastnosti potrebné pre špecifikáciu prenosu v reálnom čase, je potrebné ho doplniť ďalším protokolom, ktorým je *RTP*.

5 User Datagram Protocol (UDP) [2]

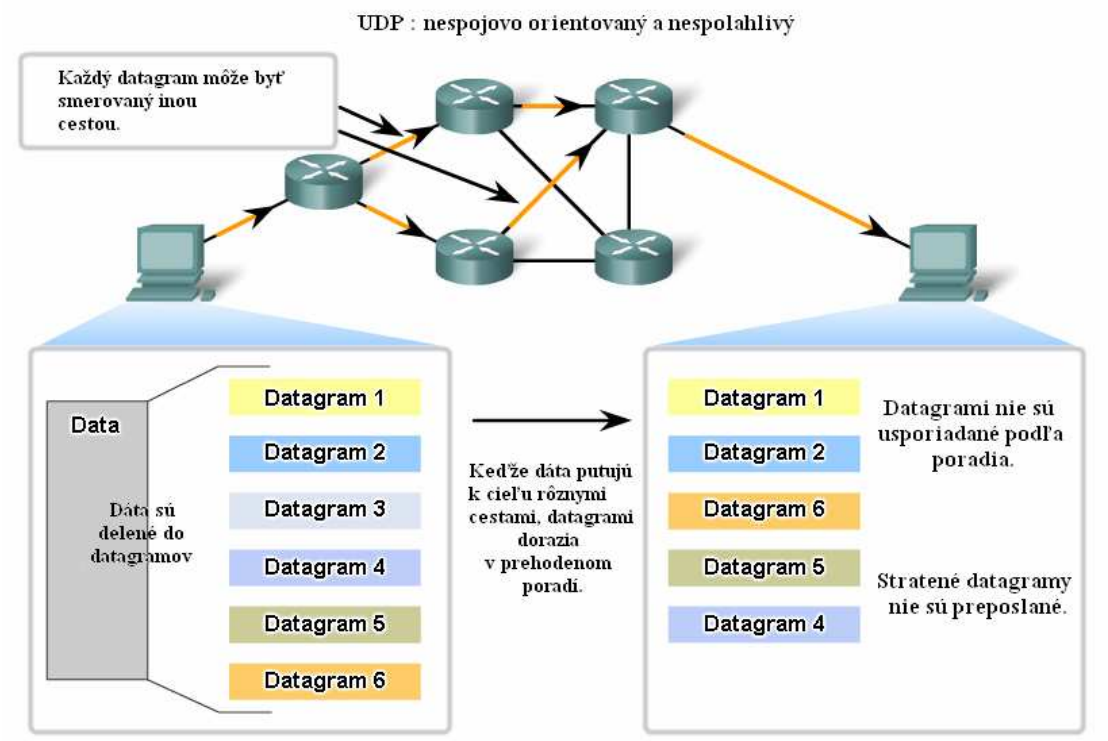
UDP je jednoduchý, nespojovo orientovaný protokol, ktorý poskytuje základné funkcie transportnej vrstvy (ISO/OSI). Tento protokol prenáša tzv. *datagramy* medzi zariadeniami v sieti a nezaručuje (ako protokol TCP), že sa prenášaný paket nestratí, že sa nezmení poradie prijímaných paketov, ani že sa niektorý paket nedoručí viac krát.
Toto neznamená, že aplikácie, ktoré používajú UDP sú vždy nespoľahlivé. Jednoducho povedané, tieto funkcie nie sú podporované transportnou vrstvou a musia byť zahrnuté niekde inde ak je to vyžadované. UDP je preto vhodný pre nenáročné a časovo citlivé účely.

Aplikácie, ktoré využívajú protokol UDP sú:

- Domain Name System (DNS)
- Simple Network Management Protocol (SNMP)
- Dynamic Host Configuration Protocol (DHCP)
- Routing Information Protocol (RIP)
- Trivial File Transfer Protocol (TFTP)

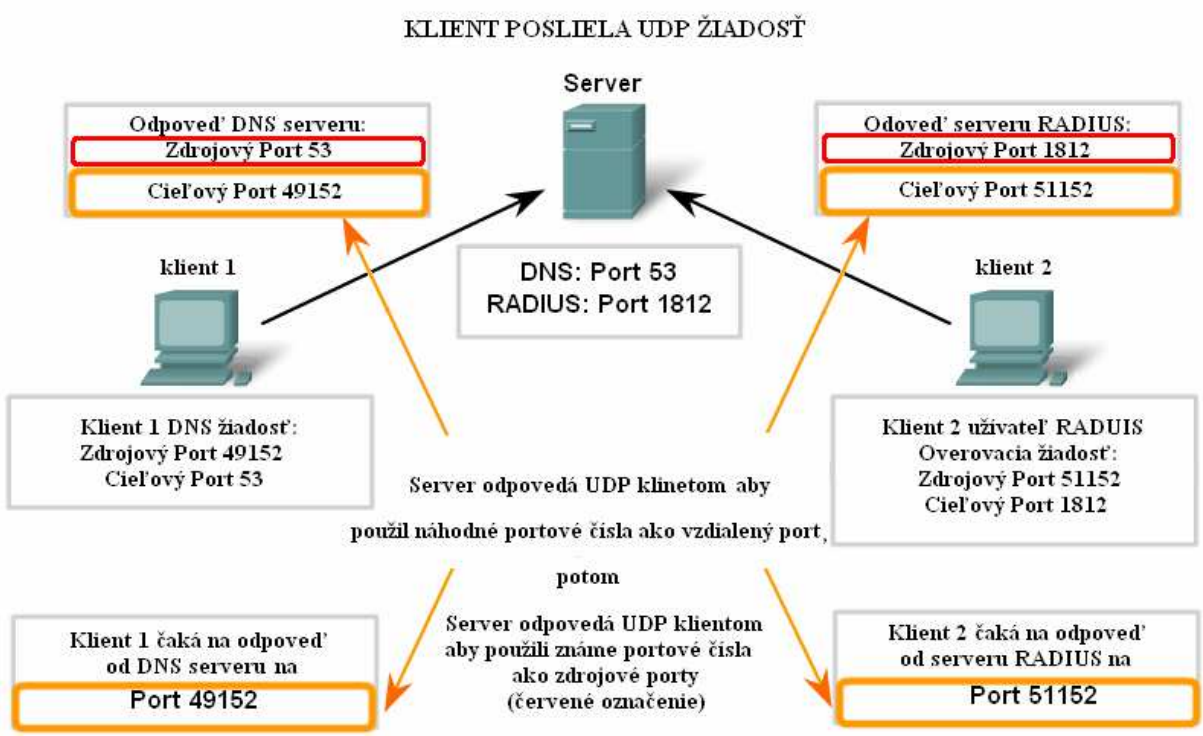
- Online hry
- Video Streaming
- Voice over IP (VoIP)

Niektoré aplikácie ako online hry alebo VoIP, môžu tolerovať stratu niektorých dát. Ak by tieto aplikácie používal protokol TCP, mohli by nastať veľké oneskorenia, pokým by TCP detekovalo stratu dát a ich preposlanie. Toto oneskorenie by mohlo byť oveľa viac škodlivé pre aplikácie ako menšie straty dát. Niektoré aplikácie ako DNS budú jednoducho opakovať požiadavku pokým im nebude doručená odpoveď. Z toho dôvodu nepotrebujú TCP záruku doručenia správy. Preto je nespojovo orientovaný protokol UDP tak žiaduci pre niektoré typy aplikácii.



Obr. 4.: Prenos datagramov po dátovej sieti

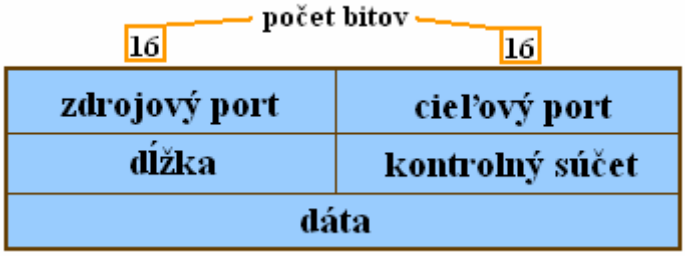
Serverové aplikácie založené na UDP sú považované ako *dobre známe,* alebo *registrované* portové čísla. Ak tieto aplikácie alebo procesy fungujú, budú prijímať dáta odpovedajúce za príslušný port. Ak UDP obdrží datagram smerovaný k jednému z troch portov, bude ich smerovať k dátovej aplikácii s príslušným portový číslom. Kapitola 8 (Porty) je vysvetlená neskôr.



Obr. 5.: Simulácia použitia UDP protokolu

5.1 Formát segmentu UDP

UDP má jednoduchý formát segmentu a je označovaný ako datagram (obr. 6) [4].



Obr. 6.: Formát segmentu protokolu UDP

Záhlavie protokolu UDP sa skladá z nasledujúcich polí [4]:

- **zdrojový port** - identifikuje zdrojový proces
- **cieľový port** - identifikuje cieľový aplikačný proces
- **dĺžka** - dĺžka celého segmentu v násobkoch počtu 32 bitov
- **kontrolný súčet** - zabezpečenie cez celý segment vrátane tzv. *pseudozáhlavia*

5.2 *Pseudozáhlavie*

Používa sa pre zabezpečenie segmentu UDP. Obsahuje niektoré informácie z IP datagramu ako: zdrojovú a cieľovú IP adresu, číslo protokolu a dĺžku pôvodného segmentu (obr. 7). *Pseudozáhlavie* slúži pre účel výpočtu *kontrolného súčtu* fiktívne pridaného pred vlastné záhlavie (ale neposiela sa spolu so segmentom). Pole *dĺžka* segmentu však obsahuje iba dĺžku pôvodného segmentu UDP a nezahrňuje pseudozáhlavie [4].

zdrojová IP adresa		
cieľová IP adresa		
0.. ..0	číslo protokolu	dĺžka segmentu

Obr. 7.: Pseudozáhlavie segmentu UDP

6 Real-Time Transport Protocol (RTP) [4]

Real-time Transport Protocol (RTP) je určený pre paketovanie toku hlasu, textu a obrazu v reálnom čase. Protokol sám neposkytuje žiadny mechanizmus na zaistenie doručenia datagramov, ich včasného doručenia a ani doručenia datagramov v správnom poradí. Definuje však poradové čísla datagramov, podľa ktorých môžu multimediálne aplikácie rozpoznať, ktorý datagram chýba. Úlohou RTP je synchronizácia časového prenosu a zaistenie straty alebo správneho poradia dát. RTP najčastejšie pracuje nad UDP (čísla portov 5004, 5005, 6970), ale môže využívať i iné protokoly.

6.1 *Štruktúra paketu RTP*

+ bity	0-1	2	3	4-7	8	9-15	16-31
0	Ver.	P	X	CC	M	PT	Sek. číslo
32	časová značka						
64	SSRC identifikátor						
96	... CSRC identfikátor ...						
96+(CCx32)	rozšírená hlavička (voliteľná)						
96+(CCx32) + (X+((EHL+1)x32))	dáta						

Obr. 8.: Štruktúra RTP paketu

RTP hlavička má veľkosť 12 bajtov a obsahuje:

- **Ver.** (2 bity) obsahuje verziu protokolu
- **P** (1 bit) používa sa pre označenie pri vyskytne extra PAD bytov na konci RTP paketu
- **X** (1 bit) označuje či sú rozšírenia protokolu použité v pakete
- **CC** (4 bity) obsahuje číslo identifikátoru CSRC, ktorý nasleduje po hlavičke

- **M** (1 bit) používa stupeň aplikácie a je definovaný jeho interpretáciou v aplikácii
- **SSRC** indikuje synchronizáciu zdroja
- **rozšírená hlavička** označuje dĺžku rozšírenia (EHL – extension header lenght) v 32 bitovej jednotke, neobsahujúcich 32 bitov rozšírenej hlavičky

7 Real-Time Transport Control Protocol (RTCP) [4]

Doručovanie paketov je monitorované pomocou podpriemerného riadiaceho protokolu *RTCP*, ktorý vytvára spätnú väzbu medzi účastníkmi relácie protokolu RTP, v ktorej periodicky prebieha výmena RTCP paketov. RTCP pakety obsahujú informácie, podľa ktorých môže vysielacia strana multimediálny tok dynamicky meniť, napr. rýchlosť prenosu, typ záťaže na základe požiadaviek prijímacej strany. Monitorovanie pomáha príjemcovi detekovať stratu paketov a uskutočniť kompenzáciu kolísania oneskorenia v sieti. Protokol RTCP tak poskytuje služby riadenia toku a kontroly zahltenia siete. RTCP používa UDP port o jedničku vyšší ako používa RTP.

8 Porty [2]

Internet Assigned Numbers Authority (IANA) stanovila portové čísla. IANA je štandard, ktorý je zodpovedný za priraďovanie rôznych adresných štandardov.

8.1 Rozdelenie portou

Dobre známe porty (čísla od 0 do 1023) – Tieto čísla sú rezervované pre služby a aplikácie. Spoločne používajú aplikácie ako HTTP (web server) POP3/SMTP (e-mailové servery) a Telnet. Definovaním *dobre známych portov* pre serverové aplikácie, môžu byť klientské aplikácie programované pre žiadosť o pripojenie špecifického portu a sním súvisiace služby.

Registrované porty (čísla od 1024 do 49151) – Tieto porty sú určené pre užívateľské procesy a aplikácie. Tieto procesy sú základné individuálne aplikácie, ktoré si užívateľ vybral nainštalovať, lepšie povedané spoločné aplikácie, ktoré budú prijímané *dobre známym portom*.
Ak nie sú použité zdrojové servery, môžu byť taktiež tieto porty vybrané dynamicky klientom ako jeho zdrojový port.

Dynamické alebo súkromné porty (čísla od 49152 do 65535) – Taktiež známe ako *krátko trvajúce porty*, sú zvyčajne určené dynamicky klientovým aplikáciám ak je naviazané spojenie. Pripojenie služieb použitím dynamických a súkromných portov nebýva veľmi zvyčajné pre klienta (aj keď programy peer-to-peer zdieľajúce zložky to podporujú).

8.2 Použitie portov pre TCP a UDP

Niektoré aplikácie môžu použiť oba protokoly TCP aj UDP. Ako príklad môžeme uviesť DNS (Domain Name System), ktorý používa UDP protokol, aby mohol veľmi rýchlo odpovedať na žiadosti klientov. Niekedy však posielanie žiadostí vyžaduje spoľahlivosť TCP protokolu. Preto v tomto prípade *dobre známy port* 53 používa oba protokoly.

Tab. 3.: Rozdelenie portov pre protokoly TCP a UDP

protokol	***dobre známe porty***	***registrované porty***	***dynamické porty***
TCP	21FTP		49152 – 65535
	23 Telnet		
	25 SMTP	1863 MSN Messenger	
	80 HTTP	8008 náhradné HTTP	
	110 POP3	8080 náhradné HTTP	
	194 IRC		
	443 HTTPS		
TCP/UDP	53 DNS	1433 MS SQL	
	161 SNMP	2948 WAP (MMS)	
	531 AOL, IRC		
UDP		1812 RADIUS	
	69 TFTP	2000 Cisco SCCP	
	520 RIP	5004 RTP	
		5060 SIP (VoIP)	

9 QoS v IP [3]

Jednou z najdôležitejších funkcií, ktorú musí sieťová infraštruktúra poskytovať svojim hlasovým koncovým zariadeniam, je správne nastavenie QoS (Quality of Services). Zložitosť celej úlohy je znásobená veľkým počtom IP telefónov. QoS je základná schopnosť siete dodržať prostredníctvom prepínačov a smerovačov v spolupráci s telefónnymi a koncovými video zariadeniami také parametry prenosu, aby bola zaistená kvalita hlasu a videa bez ohľadu na vyťažiteľnosť siete.

K riešeniu tejto úlohy boli vyvinuté nástroje umožňujúce rýchlo a automaticky konfigurovať stovky prepínačov a smerovačov. Tieto funkcie samočinne vykonávajú úlohy, ktoré sa tradične uskutočňovali ručne, ako je klasifikácia aplikácii, vytváranie pravidiel, konfigurácia QoS, monitorovanie a reporting prenosu umožňujúci testovať efektívnosť QoS a zaistiť dostatočné úrovne služieb.

9.1 Prenos sieťovým protokolom IP

Datagramový prenos prostredníctvom sieťového protokolu IP má jedinú úroveň služby: *best effort.* Internet a TCP/IP musia garantovať pre prenos aby: datagram bol prenesený čo najlepším spôsobom v závislosti na dostupných prostriedkoch a nebol akokoľvek umelo oneskorený a aby ani nedochádzalo k jeho nie celkom nevyhnutným stratám. Inými slovami povedané, všetok bežný prenos IP sa realizuje bez zaistenia doručenia datagramov do určitej doby (s minimalizáciou oneskorenia spôsobeného prenosom cez sieť) a bez zaistenia určitej šírky pásma po dobu trvania daného prenosu.

Niektoré nové typy prenosu dát v sieti ako video v reálnom čase alebo hlas, sú citlivé na odchýlky v oneskorení a na stratu paketov. Každý druh prenosu taktiež potrebuje inú šírku pásma. Preto rôzne kategórie prenosu potrebujú garanciu rôznych veličín (v určitých hodnotách) od siete vo forme *kvality služby* QoS (Quality of Services).

9.2 Metriky QoS

Medzi ***metriky QoS*** patria:

- **koncové oneskorenia** - doba medzi vyslaním paketu od zdroja k príjemcovi
- **kolísanie oneskorenia** - *jitter* - rozdiel v intervaloch medzi prijímanými paketami
- **strata paketov** - podiel prijatých a vyslaných paketov za jednotku času
- **šírka pásma** - *prenosová kapacita* - súvisiaca s priepustnosťou tj. objem dát úspešne prenesených za jednotku času

9.3 Typy Služieb ToS

Aj keď IP ponúka niekoľko *typov služieb ToS* špecifikovaných (voliteľne) priamo v datagrame, ktoré môžu znamenať rôzne zaobchádzanie s datagramom (napr. z hľadiska minimalizácie oneskorenia na ceste sieťou), ale tie sa prakticky nepoužívali kvôli zvýšeným nárokom na spracovanie najmä v smerovačoch.

Ďalšie možnosti sú *hodnoty priorít* (*IP Precedence*), ktoré prostredníctvom 3 bitov umožňovali roztriediť prenos iba do ôsmich tried služieb.

9.4 IP Precedence

IP Precedence bolo účinne nahradené DSCP (*Differential Service Code Point*), ktorý využíva šiestich bitov a umožňuje 64 tried v rámci mechanizmu diferencovaných služieb (DiffServ). DSCP ako zatiaľ najmodernejšie riešenie podpory QoS v IP sa doporučuje využiť všade, kde je to len trochu možné.

Pokiaľ má sieť splňovať požiadavky na kvalitu služby rôznych typov prenosu, musí systém pokryť niekoľko funkcii:

- **mapovanie QoS** - typicky medzi informáciou o požadovanom QoS v IP datagrame na informácie používané vo WAN
- **riadenie prístupu k médiu** - či je vôbec sieť schopná danému požiadavku vyhovieť
- **prideľovanie sieťových prostriedkov** - typicky dostupné šírky pásma

Poskytovanie sieťových prostriedkov sa rieši buď pomocou predbežnej *rezervácie* sieťových prostriedkov (signalizácia – protokol RSVP) alebo *uprednosnením prenosu* (na základe politiky).

Na základe *politiky* (súboru pravidiel) je prenos klasifikovaný do rôznych tried s podobnými požiadavkami na sieť (profily QoS). Klasifikovať prenos možno podľa rôznych hodnôt:

- *adries MAC*
- *fyzického portu prepínača* (v podnikových sieťach umožňuje koncovú QoS pre určitú skupinu užívateľov alebo špecifikované zariadenie)
- *802.1Q/p*
- *adresy IP* či *označenie aplikácie* (tj. čísel portov TCP/UDP)

Špecifikáciou vyšších protokolov možno jednotlivé aplikácie presne roztriediť, najmä pokiaľ používajú pevne pridelené, známe čísla portov. Tento spôsob sa dobre uplatnil najmä pre kritické aplikácie citlivé na oneskorenie alebo straty paketov.

9.5 DiffServ

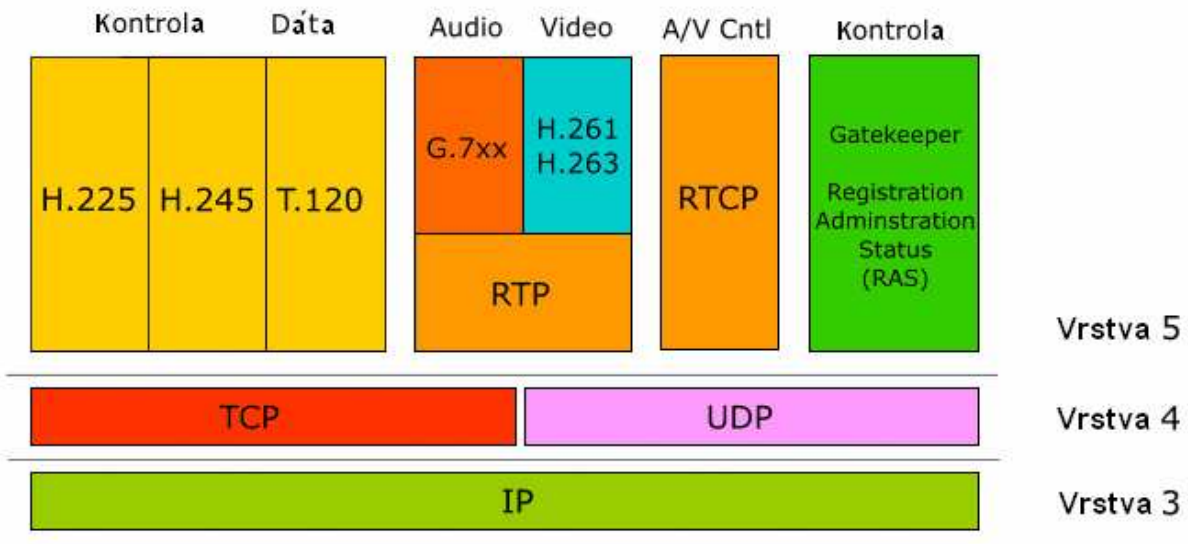
Mechanizmus *diferencovaných služieb* (*DiffServ*) využíva bity DSCP. Boli špecifikované tri základné kategórie chovania sa v skokovom rámci (PHB - Per Hop Behavior), teda medzi dvomi susednými uzlami:

- **urýchlené predávanie** (EF – Expedited Forwarding)
- **zaistené predávanie** (AF – Assured Forwarding)
- **základné služby** (BE – Best Effort)

Mechanizmus *DiffServ* sa v porovnaní s RSVP používa tam, kde je problém so škálovaním, pretože *DiffServ* agreguje jednotlivé toky dát do jedného, a preto nie je potrebné ošetrovať každý jednotlivý tok zvlášť.

10 H.323 [3]

Súbor protokolov **H.323** (*Packet-based multimedia communications systems*) využíva existujúce doporúčenie *ITU-T H.320* a *Q.931* a je určený pre konverziu signalizácie paketového protokolu na formát signalizácie telefónnej siete. Ponúka riadiaci mechanizmus pre naviazanie a ukončenie spojenia a uskutočňuje digitalizáciu a paketizáciu audio formátu pre prenos hlasu IP sieťami (obr. 9).



Obr. 9.: Architektúra H.323 v IP sieti

H.323 zahrňuje *kódovanie a kompresiu zvuku* (podľa G.7xx), *signalizáciu volania RAS* (Registration, Admission, Status) podľa *H.255* pre nadviazanie spojenia a *zabezpečenie komunikácie* medzi koncovými bodmi a *riadiacou signalizáciou* (H.245) pre každé volanie pre vzájomnú informáciu o vlastnostiach koncových bodov, otvorenie/uzatvorenie logických kanálov pre prenos, riadenie toku dát, všeobecné príkazy a indikácie stavu.

Signalizácia volania (*call signaling*) *H.255*, pracujúca nad TCP, slúži pre nadviazanie spojenia medzi dvomi koncovými bodmi a na základe výmeny správ po signalizačnom kanále, a to buď priamo medzi koncovými zariadeniami alebo medzi koncovým zariadením a zariadením *gatekeeper*. Protokol *RAS* zabezpečuje komunikáciu medzi koncovými zariadeniami (terminál, brána) a zaradením *gatekeeper*.

H.323 definuje štyri základné funkcie:

- **Terminál**
- **bránu**
- **gatekeeper**
- **MCU**

Terminál je základná a povinná zložka pre obojsmernú komunikáciu v reálnom čase. Musí podporovať H.245 pre vytvorenie prenosového kanálu pre zvuk a video; H.255 pre signalizáciu; RAS pre prípadnú komunikáciu s gatekeeper; RTP/RTCP pre zaistenie správneho poradia audio a video paketu; G.72x pre kódovanie zvuku.

Brána zabezpečuje obojsmernú komunikáciu terminálov v sieti s *H.323* s terminálmi v iných sieťach (napr. ISDN, SS7) a slúži ako prekladač protokolov. Na strane siete *H.323* pracujú protokoly *H.245, H.255 a H.255 RAS* pre registráciu *gatekeeper*. ***Gatekeeper*** poskytuje riadiace služby pre terminály a brány a nepodieľa sa na prenose samotných dát (hlasu). Ako voliteľná zložka podporuje riadenie prístupu, adresovanie, účtovanie, manažment šírky pásma, riadenie volania s ohľadom na jeho požiadavky na sieť a signalizáciu, smerovanie hovorov apod. Pre riadenie brány sa používa protokol *Megaco/H.248* a *H.248.1*, ktoré definujú komunikáciu medzi *media gateway* a *MGC* (*Media Gateway Controller*) pre rozdelenú bránu.

MCU (*Multipoint Control Unit*) je riadiaca jednotka pre komunikáciu viacej terminálov a brán (v praxi býva v kombinácii s bránou a gatekeeper v jednom zariadení).

Jedným s problematických miest je závislosť *H.323* na spoľahlivom transportnom protokole *TCP*, pretože udržovanie veľkého počtu relácii *TCP* vedie k značnej réžii v sieti. Pri naviazaní spojenia sa nastaví relácia *TCP* pomocou protokolu *H.255.0* s využitím zapúzdrenia správ podľa *Q.931*. Táto relácia sa udržuje po celú dobu trvania spojenia. Pre dojednanie možností komunikačných strán, určenie vzájomných vzťahov (*master-slave*) a vysielanie toku dát sa používa druhá relácia *TCP* s použitím protokolu *H.245*.

H.323 sa nezaoberá mechanizmom pre zabezpečenie šírky pásma. Behom naväzovania spojenia si koncový účastníci môžu v rámci signalizácie *H.255 RAS* vymeniť správu o svojej schopnosti podporovať *QoS*. Pre vlastnú rezerváciu pásma sa používa protokol *RSVP*, ktorý má problémy so škálovaním, lebo uskutočňuje manažment na úrovni jednotlivých tokov aplikačných dát.

11 H.263 [9]

H.263 je video kodek štandardu ITU-T navrhnutého v projekte na prelome rokov 1995/1996 ako nízko bitová prenosová rýchlosť komprimovaného formátu video sekvencie. Patrí do skupiny štandardov video kódovania H.26x v oblasti ITU-T Video Coding Experts Group (VCEG).

H.263 bol vyvinutý ako vývojové zlepšenie predchádzajúceho štandardu ITU-T (kodek H.261) pre video kompresiu. Kodek H.263 bol naďalej vyvíjaný a rozšírený na verziu známu ako H.263v2 a neskôr H.263v3 (alebo H.263++).

11.1 Použitie

H.263 video môže byť dekódované s LGPL- licencovaný libav-kodek knižnicou, ktorá je použitá v programoch ako ffdshow, VLC media player a Mplayer. Originálna verzia kodeku RealVideo je založená na H.263.

11.2 Rozdieli medzi jednotlivými verziami H.263 [8]

	verzia 1	verzia 2	verzia 3
pokročilé INTRA kódovanie	Áno	Áno	Áno
filter deblokácie	Áno	Áno	Áno
pomocné zvýšenie informácii	Áno	Áno	Áno
upravené kvantovanie	Áno	Áno	Áno
neobmedzený pohyb vektorov	Nie	Áno	Áno
mód štrukturovania časti	Nie	Áno	Áno
znovu-opätovné poskladanie obrázku	Nie	Áno	Áno
rozšírená predvídateľnosť	Nie	Nie	Áno
zlepšenie PB-frejmov	Nie	Nie	Áno
nezávislé segmentové dekódovanie	Nie	Nie	Áno
alternatívne INTER VLC	Nie	Nie	Áno
	verzia 1	verzia 2	verzia 3

Obr. 10.: Porovnanie funkcii jednotlivých verzii H.263

12 Network Simulator verzie 2

12.1 Úvod

Network Simulator verzie 2 sa využíva pre simuláciu dátových sietí. Podporuje protokoly ako UDP, TCP a iné smerovacie a multikástové protokoly. Tento simulátor bol vyvinutý v roku 1989 a od roku 1995 bol podporovaný DARPou (DARPA-počiatky internetu). Jedná sa o otvorený systém, ktorý umožňuje akékoľvek zmeny zadané užívateľom. Tým je ale aj daná jeho zložitosť. Network Simulator je napísaný v programovacom jazyku C++ a v skriptovacom jazyku Otcl. Programy na vytvorenie simulácii sú písané v textových súboroch a majú koncovú príponu „***.tcl**“. Aby bol možný grafický výstup, je potrebný ďalší program a tým je NAM, ktorý umožňuje grafické zobrazenie topológie siete naprogramovanej za pomoci NSv2. Za to program TraceGraph umožňuje grafický výstup. Network Simulator je voľne šíriteľný program a je možné si ho stiahnuť na stránke http://www.isi.edu/nsnam/ns/ , kde sú uvedené aj potrebné informácie k jeho inštalácii [10].

12.2 Inštalácia simulátoru NSv2

Network Simulator verzie 2 bol navrhnutý pre unixové platformy a je dostupný na stránke http://www.isi.edu/nsnam/ns/ns-build.html . Pre inštaláciu je potrebný program „Cygwin“, ktorý poskytuje Linuxové rozhranie pre prácu pod Windowsom. Jeho stiahnutie je možné na stránke http://www.cygwin.com/ . Po stiahnutí súboru „cygwin.exe“ a následnom inštalovaní je potrebné označenie knižníc, ktoré sa majú do Cygwin nainštalovať (ak to nespravíme, nebudeme môcť NSv2 úspešne nainštalovať) [10].

Po stiahnutí balíku súborov pre NSv2 je potreba tento balík rozbaliť a do Cygwin konzole napísať *./install* .

Ak sa vyskytne chyba, riešenie nájdeme na http://www.isi.edu/nsnam/nam/nam-problems.html, alebo chybou môže byť, že sme nenainštalovali všetky potrebné knižnice pri inštalácii Cygwin (teda inštaláciu Cygwin je potrebné znovu zopakovať).

TraceGraph je dostupný na http://www.tracegraph.com/.

Po inštalácii sa musia nastaviť systémové premenné $PATH, $LD_LIBRARY_PATH, $TCL_LIBRARY, $TK_LIBRARY pripísaním týchto riadkov do *.BASHRC:*

export PATH=$PATH:/home/<uzivatel>/ns-allinone-2.29/bin:/home/<uzivatel>/ns-allinone-2.29/tcl8.4.11/unix:/home/<uzivatel>/ns-allinone-2.29/nam-1.11:/home/<uzivatel>/tracegraph

export LD_LIBRARY_PATH=/home/<uzivatel>/ns-allinone-2.29/otcl-1.11:/home/<uzivatel>/ns-allinone-2.29/lib:/home/<uzivatel>/matlab/bin/glnx86

export TCL_LIBRARY=/usr/share/tcl/tcl8.4:/home/<uzivatel>/ns-allinone-2.29/tcl8.4.11 /library

export TK_LIBRARY=/usr/share/tk8.4

Pokiaľ máme funkčnú inštaláciu NSv2, potom samotné programovanie sa robí v ľubovolnom textovom editore. Výsledný súbor musí mať príponu ***.tcl**. Pre spustenie simulácie musí pracovný adresár obsahovať ***.tcl** súbor. Spúšťanie sa urobí príkazom **$startx** (v konzolovom okne Cygwin). Po tomto príkaze sa otvorí nové okno, v ktorom je najprv potrebné zadať cestu k cieľovému adresáru (<uzivatel> /ns/bin) a až potom je možné spustiť vytvorený program za pomoci príkazu ***./ns priklad.tcl***. V zdrojovom kóde sú už zahrnuté odkazy na *NAM* a *TraceGraph.*

13 Simulácia v programe NSv2

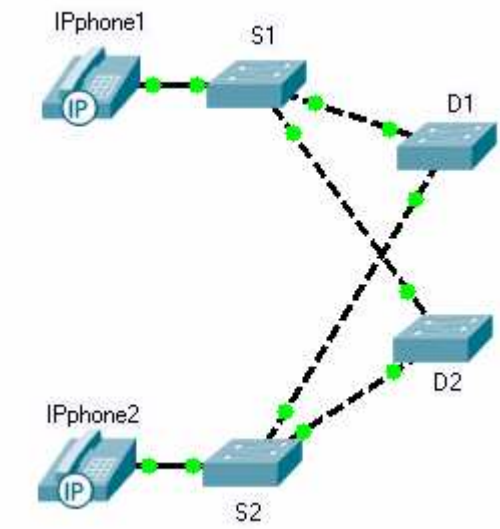
13.1 Zadanie

Simulujte prenos hlasu a videa v konvergovaných sieťach so zameraním na QoS. V prostredí Network Simulator verzie 2 vytvorte vhodnú topológiu IP siete. Vytvorte najprv paketový hlasový tok medzi účastníkmi, uvažujte kompresiu podľa doporučenia G.723. Analyzujte oneskorenie, jitter a stratovosť paketov hlasového toku. Ďalej vytvorte prenos videa v reálnom čase s uvážením doporučenia H.263. Simulujte poruchu linky, sledujte dopad na parametre linky a navrhnite možné odstránenie tejto poruchy.

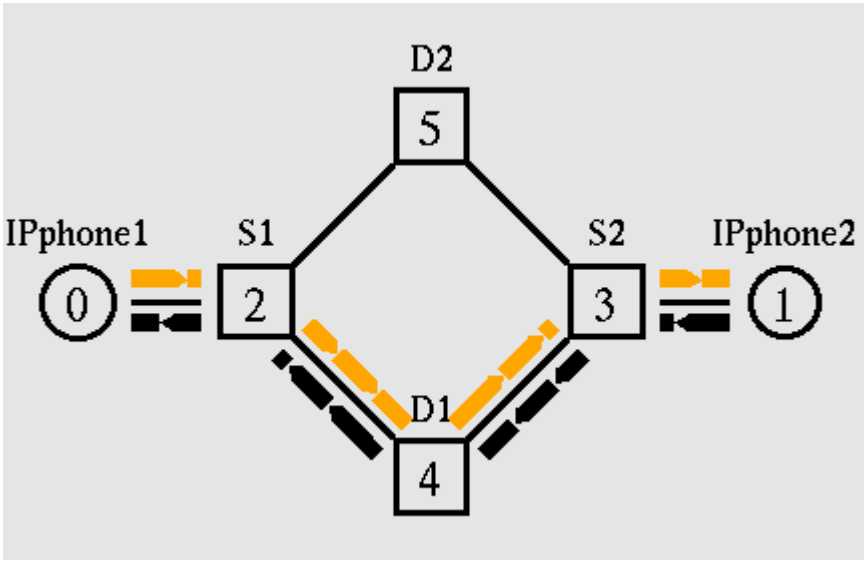
13.2 Vytvorenie IP siete pre simuláciu prenosu hlasového toku

13.2.1 Navrhnutie IP siete

Pomocou skriptovacieho jazyku Otcl som vytvoril IP sieť s vhodnou topológiou pre simuláciu prenosu hlasového toku. Na obr. 11 je zobrazená topológia siete a na obr. 12 je zobrazenie za pomoci programu NAM .



Obr. 11.: Topológia IP siete pre simuláciu prenosu hlasového toku



Obr. 12.: Topológia IP siete pre simuláciu prenosu hlasového toku zobrazená za pomoci NAM

Na tomto príklade som simuloval IP hovor, pri ktorom nastáva komunikácia medzi užívateľmi IPphone1 a IPphone2 (čierna a oranžová). Keďže sa jedná a obojsmernú komunikáciu (full-duplex komunikáciu), sú vysielané pakety od oboch užívateľov súčastne.

Pre správne fungovanie hovorov musia protokoly zabezpečiť nasledujúce body:

- nastavenie hovoru
- udržanie hovoru
- zrušenie hovoru

13.2.1.1 Nastavenie hovoru [11]

Pri *nastavení hovoru* sa overuje konfigurácia smerovania hovoru pre stanovenie cieľa volania. Konfigurácia určuje šírku pásma, ktorá je potrebná pre prenos hovoru (literatúra uvádza najmenej 128 kbps pre full-duplex komunikáciu). CAC (Call Admission Control) definuje či je dostupná dostatočná šírka pásma pre hovor. Ak áno, zaháji sa vysielanie signálov. Odlišné protokoly pre nastavenie hovoru ako H.323, MGCP a SIP definujú odlišné nastavenia pre zostavenie hovoru. Avšak všetky protokoly nastavujú rovnaké základné informácie:

- IP adresu obidvoch zariadení, ktoré si budú vymieňať VoIP
- UDP portové čísla, ktoré budú použité pre RTP streamy prenášajúce hlasové dáta
- formát použitý pre digitalizovanie hlasu

13.2.1.2 Udržanie hovoru [11]

Pri *udržaní hovoru* sa sleduje počet paketov, stratené pakety, jitter a oneskorenie počas prebiehania hovoru. Informácie prechádzajúce zariadením (IPtelefón, počítač..) určujú aká je kvalita spojenia. Ak je kvalita zhoršená, je potrebné určiť kde gateway preruší hovor.

13.2.1.3 Zrušenie hovoru [11]

Zrušenie hovoru oznamuje zariadeniam (IPtelefón, počítač..) možnosť dostupu iných hovorov a učiní ich dostupnými pre iné hovory ak jedna alebo druhá strana ukončí hovor.

13.2.2 Navrhnutie IP siete (pokračovanie)

V simulácii som použil dve skupiny switchov: access(S) a distribution(D), ktoré zabezpečujú efektívnejšie využitie šírky pásma siete. Switche typu S sú prístupové (access) a sú napojené na rozvodové (distribution) switche. Topológiu siete som navrhol tak, aby bola možná komunikácia aj pri nastaní poruchy. IP telefóny by mohli byť pri tejto simulácii nahradené počítačmi s príslušným softvérom pre IP telefóniu.

13.2.3 Určenie veľkosti paketov pre VoIP [12]

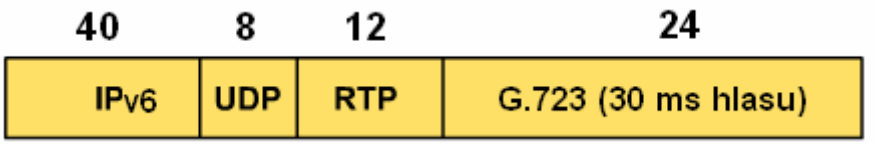
Veľkosť VoIP paketu záleží na kodeku, ktorý je použitý a na množstve hlasu, ktorý je paketizovaný. Dôležitými charakteristikami kodeku sú:

- počet bitov vyprodukovaných za sekundu
- perióda – ako často sú vzorky vysielané

Zo zadania je dané, že mám vytvoriť paketový hlasový tok medzi účastníkmi, s použitím kompresie podľa doporučenia G.723. Jedná sa o hlasový kodek, ktorý využíva ADPCM moduláciu s bitovou rýchlosťou **6,4 kbps**. Tento audio kodek komprimuje hlas do **30 ms** frejmov o veľkosti **24 bajtov** [12].

Veľkosť hlasového paketu je daná nasledovne:	- G.723	24 bajtov
	- RTP	12 bajtov
	- UDP	8 bajtov
	- IPv6	40 bajtov
	- Ethernet záhlavie	38 bajtov

Teda po súčte je veľkosť paketu **122 bajtov**.



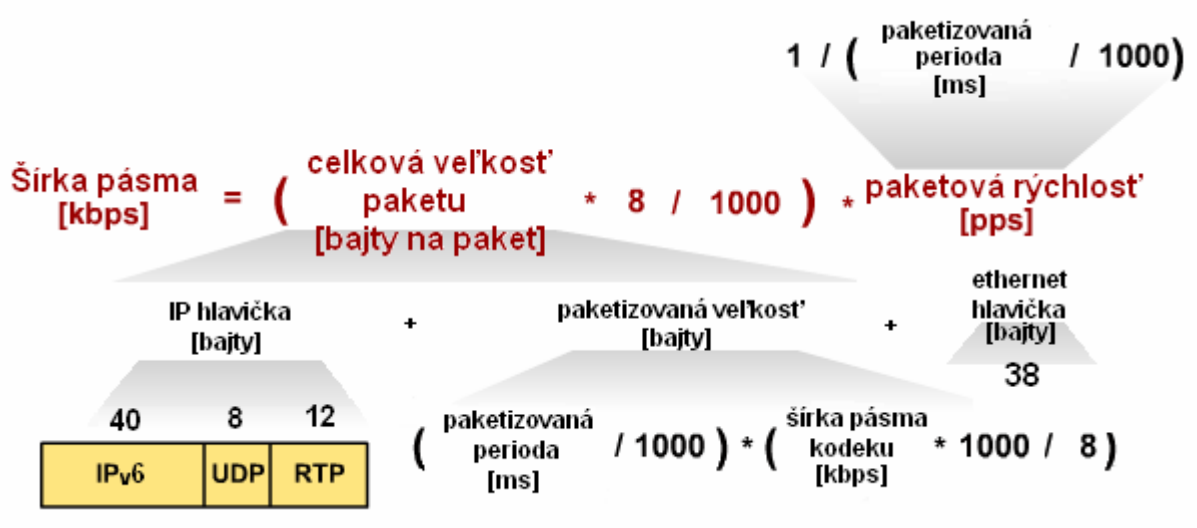
Obr. 13.: Zapúzdrenie hlasového paketu a jeho veľkosť (bez Ethernet záhlavia)

13.2.4 Príklad výpočtu šírky pásma pre VoIP

Ak navrhujeme sieť pre VoIP, je dôležité vedieť celkovú šírku pásma pre VoIP hovory. Táto informácia je dôležitá pre určenie kapacity fyzických liniek medzi uzlami a správneho rozmiestnenia QoS. QoS udáva prioritu pre hlasové pakety, predchádza vysokým oneskoreniam, ktoré sú spôsobené radením do fronty (queuing), a tým vplýva na kvalitu hlasu.

Vzorec pre výpočet celkovej veľkosti šírky pásma VoIP hovoru je nasledovný [12]:

$$Šírka\ pásma = (122 * 8 / 1000) * 33{,}3 = \mathbf{32{,}5\ kbps} \qquad (13.1)$$



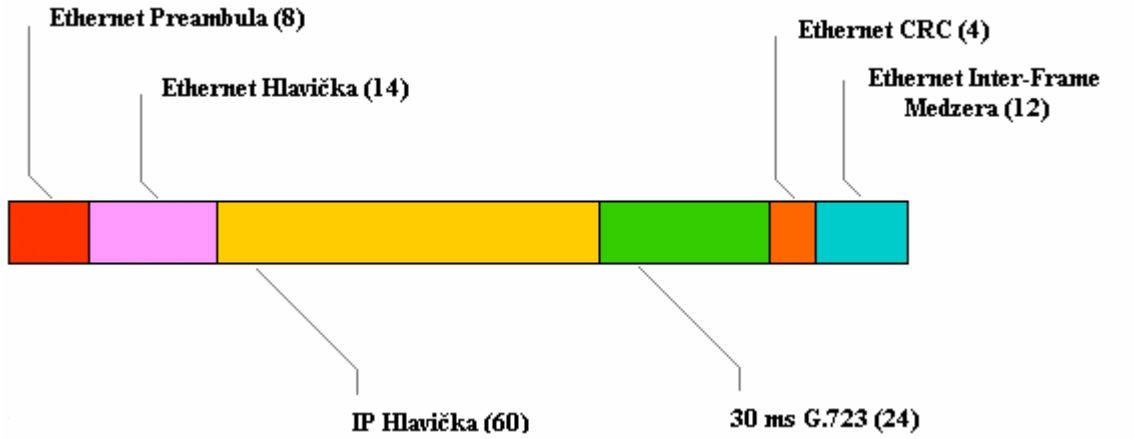
Obr. 14.: Vzorec výpočtu celkovej veľkosti šírky pásma VoIP hovoru

K šírke pásma sa môžeme dopracovať aj nasledujúcim spôsobom [12]:

Jeden paket je vysielaných každých 30 ms, to je 33,3 paketu za sekundu. Užitočnosť prenosu je 6400 ÷ 33,3 = 192 bitov (24 oktetov).

Pevná veľkosť IP hlavičky je 60 oktetov a pevná veľkosť Ethernet hlavičky je 38 oktetov Celková veľkosť paketu je teda 122 oktetov. Potom potrebná šírka pásma je vypočítaná nasledovne:

$$(24 + 60 + 38)\ x\ 33{,}3\ x\ 8 = \mathbf{32{,}5\ kbps}.$$



Obr. 15.: Zobrazenie celého hlasového paketu

Výsledok nám vyšiel iba pre jeden kanál, a keďže sú kanály dva (incoming – outgoing chanel), je potrebné vynásobiť šírku pásma dvomi:

$$32{,}5\ kbps\ x\ 2 = \mathbf{65\ kbps}$$

Tým sme dostali celkovú šírku pásma pre vysielací i prijímací kanál, ktorá sa líši od hodnoty uvedenej v literatúre (64 kbps) iba o 1 kbps. Výpočet je teda správny.

13.2.5 Jitter [14]

Jittter je rozdiel v intervaloch medzi prijímanými paketami. Závisí od bitovej rýchlosti a od veľkosti paketov. Ak je rýchlosť prenášaných dát hlasu 64 kbps, môžem preniesť 1920 bitov v 30 ms. Ak platí, že R je bitová rýchlosť a J je označenie pre najhorší jitter, platí nasledujúci vzťah:

$$veľkosť\ buffera = 2 * J * R \quad [bity] \tag{13.2}$$

po dosadení

$$veľkosť\ buffera = 2 * 30 * 64 = \mathbf{3840\ [bitov]}.$$

Vysvetlenie:

Pred zahájením prenosu, treba vyčkať na čas jitteru (30 ms), aby sa predišlo poruche, pretože prvý paket môže byť vyslaný priskoro, ale ďalší príliš neskoro. Takže je potrebný buffer s 1920 bitmi k prekonaniu diferencie. No môže nastať situácia, keď prvý paket príde oneskorene, a potom je treba vyčkať na čas jitteru (30 ms) a tak zase ďalší paket príde priskoro. K predídeniu presiahnutia bufferu je treba umiestniť ďalších 1920 bitov dát. Takže buffer bude potrebovať najmenej 3840 bitov (480 bajtov).

13.2.6 Príčiny oneskorenia paketov [13]

Procesové oneskorenie (spracovanie) d_{pro}
- skontrolovanie chybných bitov
- určenie výstupnej linky

Vysielacie oneskorenie d_{trans}
- R= šírka pásma linky (bps)
- L= dĺžka paketu (bits)
- čas vyslania bitov = L/R

Oneskorenie frontou (queueing) d_{queue}
- čas čakania kým sa bude vysielať
- závisí od zaplnenia fronty

Oneskorenie prenosom d_{prop}
- d= dĺžka fyzického spojenia
- s= prenosová rýchlosť (~$2x10^8$ m/s)
- oneskorenie prenosom = d/s

potom oneskorenie uzlu je:

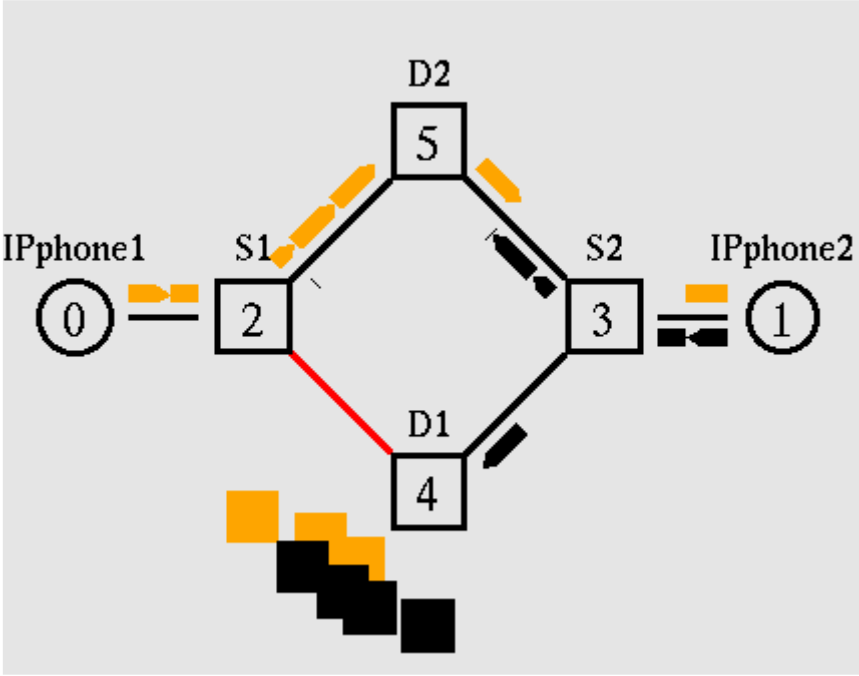
$$d_{uzlu} = d_{pro} + d_{queue} + d_{trans} + d_{prop} \tag{13.3}$$

Upozornenie: s a R sú dve odlišné veličiny !

13.2.7 Simulácia poruchy linky

Z obr. 12 je viditeľné, že hovor prichádzajúci od IPphone2 vedie cez uzly *S2->D1->S1* a hovor prichádzajúci od IPphone1 vedie cez uzly *S1->D1->S2*. Pre overenie funkčnosti siete som na linke vytvoril poruchu medzi uzlami S1 a D1. Následkom bola strata paketov, ktoré touto linkou práve prechádzali (Obr.16), ale aj vytvorenie náhradného (záložného) spojenia. Toto spojenie sa aktivovalo príkazom pre zavolanie „distance-vector“ smerovacieho protokolu, za pomoci ktorého došlo k nájdeniu spojenia a k prenosu paketov cez uzly S1->D2->S2.

K celkovému oneskoreniu je treba pričítať 30 ms, čo je doba, kedy dochádza k prevodu hlasu podľa doporučenia G.723.



Obr. 16.: Simulácia poruchy linky pre prenos hlasového toku

13.2.8 Definovanie zdrojového kódu pre vytvorenie simulácie

Pre spustenie a správne fungovanie vytvoreného simulovaného programu je potrebné do neho vložiť príkazy, kde vyvoláme smerovací protokol, nastavíme si farbu sledovaných paketov, otvoríme súbor pre zápis v NAM, definujeme procedúru ukončenia, atď. Ja som si nazval túto časť príkazov ako „hlavičku“. Bez nich by program nepracoval správne.

V nasledujúcich riadkoch sú uvedené potrebné príkazy a ich popis činnosti.

Pre vytvorenie objektu v NS použijeme nasledujúci príkaz:

```
set ns [new Simulator]
```

Pri simulácii poruchy linky je potrebné aktivovať smerovací protokol, ktorý nájde náhradné (záložné) spojenie. Učiní sa tak príkazom:

```
$ns rtproto DV
```

Aby sme mohli lepšie rozlíšiť vysielané a prijímané pakety, definujeme ich farbu nasledujúcim príkazom:

```
$ns color 1 Orange
```

Pre otvorenie súboru pre zápis v NAM (grafické zobrazenie pomocou NAM), musíme tento program vyvolať príkazmi:

```
set nf [open out.nam w]
$ns namtrace-all $nf
```

A teraz definujeme procedúru ukončenia:

```
proc finish {} {
        global ns nf
        $ns flush-trace
        close $nf
        exec nam out.nam &
        exit 0
}
```

Nasledujúcimi príkazmi sa tvorí „telo“ programu. Definujeme si uzly a ich pomenovanie, tvar a rozmiestnenie. Vytvoríme linky medzi nimi a nastavíme im veľkosť fronty. Určíme si akými protokolmi budú pakety vysielané, kto má vysielať a kto prijímať, atď.

Vytvorenie uzla, jeho pomenovanie a nastavenie tvaru (box, hexagon):

```
set uzol [$ns node]
$ns at 0.0 "$uzol label Meno_Uzla"
$ uzol shape "box"
```

Uzly spojíme tak, že vytvoríme linky medzi nimi. V príkaze je postupne definované o aké spojenie sa jedná (duplex-link), medzi akými uzlami sa vytvára spojenie (uzol_1 a uzol_2), šírka pásma medzi uzlami (1Mb), oneskorenie (10ms) a typ fronty (DropTail):

```
$ns duplex-link $uzol_1 $ uzol_2 1Mb 10ms DropTail
```

Ďalej nastavujeme veľkosť fronty medzi uzlami:

```
$ns queue-limit $ uzol_1  $ uzol_2  10
```

Pre grafické rozmiestenie uzlov sa znovu definuje o aké spojenie sa jedná (duplex-link-op), ktoré uzly spájame (uzol_1 a uzol_2) a ako sú vzájomne orientované (orient right-up):

```
$ns duplex-link-op $ uzol_1 $ uzol_2 orient right-up
```

Sledovanie stavu na linkách možno za pomoci príkazu:

```
$ns duplex-link-op $ uzol_1 $ uzol_2 queuePos 0.5
```

Ďalej si vytvoríme RTP agentov:

```
set RTP [new Agent/RTP]
$ns attach-agent $phone1 $RTP1
```

```
set sink [new Agent/RTP]
$ns attach-agent $phone2 $sink
$ns connect $RTP1 $sink
$RTP1 set fid_ 1 #Priradenie farby
$RTP1 set packetSize_ 976
```

Pre vytvorenie aplikácie generujúcej konštantný dátový tok „hovor1“ použijeme príkazy:

```
set hovor1 [new Application/Traffic/CBR]
$hovor1 attach-agent $RTP1
$hovor1 set packetSize_ 24
$hovor1 set rate_ 6.4kb
```

Pre modelovanie výpadku linky je potrebné definovať v akom čase, a ktorá linka (medzi ktorými uzlami) má mať výpadok:

```
$ns rtmodel-at 1.0 down $s1 $d1
```

Teraz si už len definujeme časovanie udalosti pre generátory prevádzky:

```
$ns at 0.0 "$RTCP1 start"
$ns at 3.0 "$RTCP1 stop"
```

Na koniec programu dopíšeme nasledujúce príkazy (tzv. „koncové“), kde definujeme ukončenie programu a odpojenie generátora a konzumenta.

Zavolanie procedúry finish po 3 sekundách od spustenia simulácie:

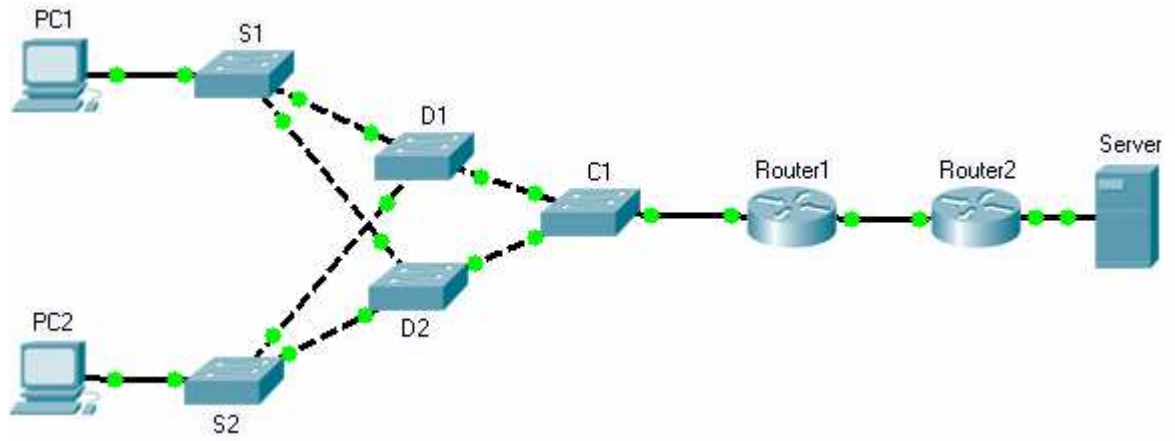
```
$ns at 3.0 "finish"
```

Ako úplne posledný príkaz napíšeme „`$ns run`“, ktorým sa vyvolá zahájenie simulácie.

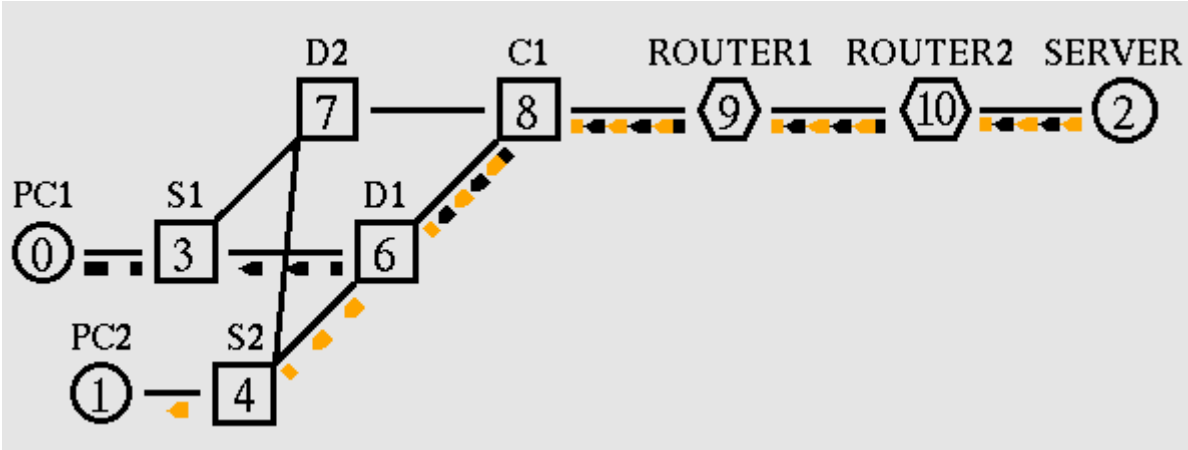
13.3 Vytvorenie IP siete pre simuláciu prenosu videa v reálnom čase

13.3.1 Navrhnutie IP siete

Takisto ako v predchádzajúcej simulácii, vytvoril som znovu IP sieť s vhodnou topológiou pre simuláciu prenosu videa v reálnom čase. Na obr. 17 je zobrazená topológia siete a na obr. 18 je zobrazenie za pomoci programu NAM..



Obr. 17.: Topológia IP siete pre simuláciu prenosu videa v reálnom čase



Obr. 18.: Topológia IP siete pre simuláciu prenosu videa v reálnom čase zobrazená za pomoci NAM

Pri tejto simulácii je v ľavej časti obrázku použitá taká istá topológiu siete ako pri prenose hlasových dát, a však pripojená je k inej sieti za pomoci routeru. Spojenie medzi sieťami (spojenie medzi routermi) zabezpečuje internetový provider. Dáta vo forme videa sú streamované na požiadanie účastníkov zo serveru na užívateľské počítače PC1 a PC2. Prenos využíva nespojovo orientovaný protokol UDP. QoS sa zachová prostredníctvom prepínačov a smerovačov tak, aby bola zaistená kvalita prenášaných dát bez ohľadu na vyťažiteľnosť siete.

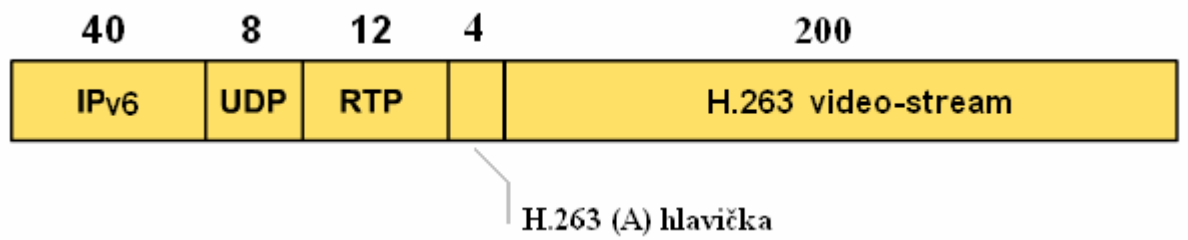
13.3.2 Určenie veľkosti paketov pre video v reálnom čase

Veľkosť video paketu taktiež záleží na kodeku, ktorý je použitý. Zo zadania je dané, že máme vytvoriť prenos videa v reálnom čase s uvážením doporučenia H.263. Jedná sa o video kodek, ktorý podporuje video kompresiu pre video-konferenciu a video-telefóniu s nízkou-bitovou kompresiou dát. H.263 je definovaný pre tri formáty hlavičiek (A, B, C). Ja som si zvolil formát typu (A), ktorý prezentuje **4 bajty** pred video streamom skomprimovaným podľa kodeku H.263. Jeho výhodou je jednoduchosť a malá veľkosť hlavičky (spomínané 4 bajty), takže umožňuje efektívnejšie spracovanie malých paketov a paketov s nízkou bit. rýchlosťou.

Veľkosť video paketu je daná nasledovne:

- H.263 (A)	4 bajty
- H.263 video-stream	200 bajtov
- RTP	12 bajtov
- UDP	8 bajtov
- IPv6	40 bajtov
- Ethernet záhlavie	38 bajtov

Teda po súčte je veľkosť paketu **302 bajtov**.



Obr. 19.: Zapúzdrenie video paketu a jeho veľkosť (bez Ethernet záhlavia)

13.3.3 Príklad výpočtu šírky pásma pre video v reálnom čase

Kodek H.263 podporuje štandardy MPEG-1 a MPEG-2. Ak teda uvažujeme, že v simulácii je použitý štandart MPEG-1 bude minimálna bitová rýchlosť **1,5 Mbps.**

Pretože:

- 1,25 Mbps video 352 x 240 x 30 Hz
- 250 kbps audio (dva kanále)

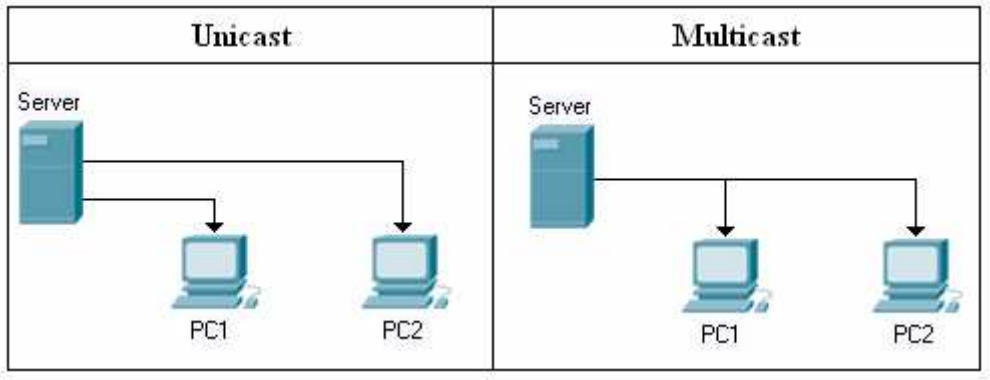
Pre výpočet celkovej šírky pásma taktiež musíme brať do úvahy, či sa jedná o prenos „Multicast“ alebo „Unicast“ (obr. 20). Pri multicast prenose je celková šírka pásma rovná bitovej rýchlosti, t.j. 1,5 Mbps. No ak by sa jednalo o unicast prenos, celková šírka pásma sa vypočíta takto:

celokvá šírka pásma [Mbps] = bitová rýchlosť [Mbps] x počet užívatelov (13.3)

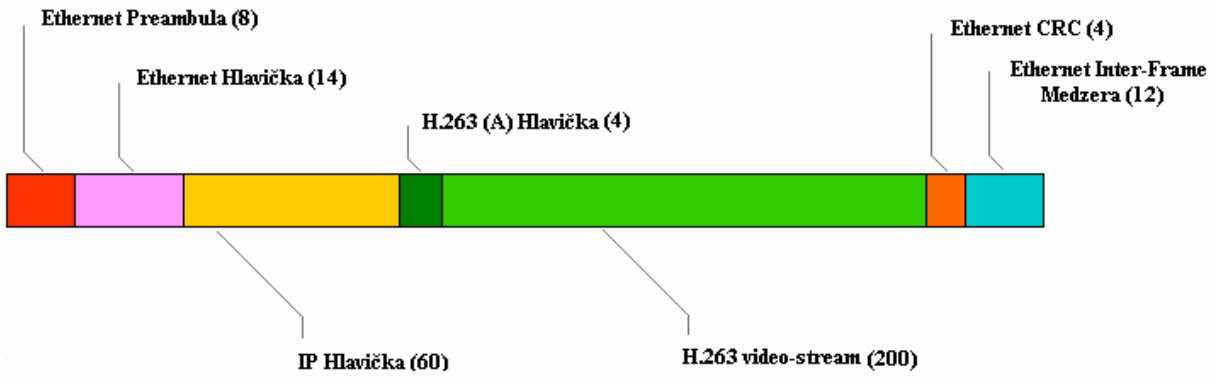
odtial potom:

celková šírka pásma [Mbps] = 1,5 Mbps x 2 = ***3 Mbps***.

Z toho vyplýva, že čím je počet užívateľov väčší, tým väčšia musí byť aj celková šírka pásma.



Obr. 20.: Rozdiel medzi „Unicast“ a „Multicast“ prenosom



Obr. 21.: Zobrazenie celého video paketu

13.3.4 Streaming

Streaming je technológia šírenia audiovizuálneho obsahu po internete. Od obyčajného posielania súbor sa líši synchrónnym prenosom dát, kedy jedna strana prijíma a druhá vysiela (klient-server). Streamovací server zaisťuje plynulé vysielanie dát a komunikáciu s cieľovými zariadeniami. Aby bola zaistená dostatočná kvalita prenosu, musí mať server kvalitnú konektivitu a byť schopný distribuovania veľkého množstva dát.

13.3.5 Kapacita pamäte pre streamované dáta [15]

Pre server, ktorý vysiela audiovizuálne dáta je potrebné vypočítať kapacitu pamäte pre streamované dáta. Tá je závislá na celkovej šírke pásma a času prenesených dát. Pre výpočet platí nasledovný vzorec:

minuta prenosu (60 s): kapacita pamäte = (60 x 1500) / 8 388,608 = **10,73 MB**.

hodina prenosu (3600 s): kapacita pamäte = (3600 x 1500) / 8 388,608 = **643,7 MB**.

$$\text{kapacita pamäte [MB]} = \frac{\text{čas [s]} \times \text{bitová rýchlosť [kbps]}}{8\,388{,}608\ \text{[kbits]}}$$

1 MB = 8 * 1 048 576 [bitov]

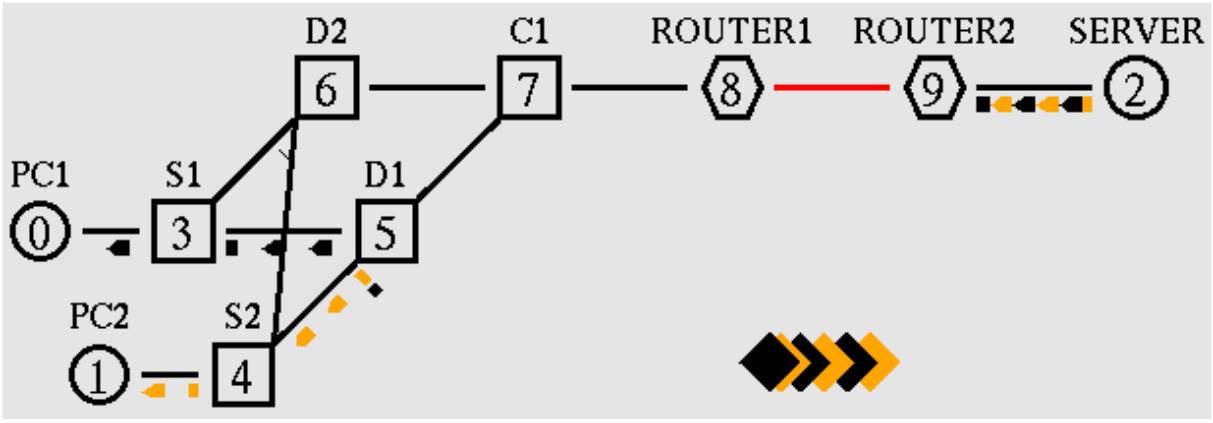
Obr. 22.: Vzorec pre výpočet kapacity pamäte pre streamované dáta

13.3.6 Simulácia poruchy linky

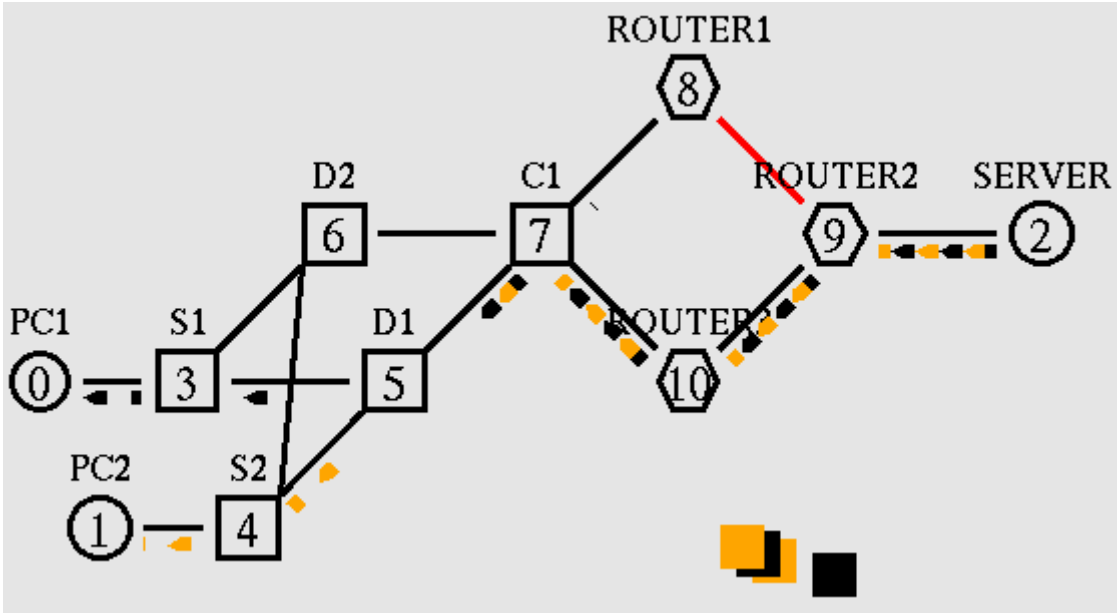
Poruchy liniek mohli u tejto simulácie nastať vo viacerých miestach. Ja som si z nich vybral dve, ktoré som ošetril, a to vytvorením záložných spojení. Prvá porucha je ošetrená medzi routermi Router1 a Router2. Druhá porucha sa nachádza medzi uzlami C1 a D1.

13.3.6.1 Simulácia poruchy linky medzi routermi R1 a R2

Pri prenose dát z jednej siete do druhej môže nastať porucha medzi routerami. A tak nie je možné vyslanie dát k počítačom PC1 a PC2 na prijímacej strane (obr. 23). Problém poruchy sa eliminuje pridaním Routeru3, ktorý bude slúžiť ako záložné spojenie. Ako je vidno z obr. 24, pri poruche medzi Routerom1 a Routerom2 nastane strata paketov, ktoré sú práve vysielané poruchovou linkou. V zápätí sa aktivuje záložné spojenie cez Router3 a pokračuje sa v prenose dát. Na prijímacej strane sa strata paketov prejaví krátkym výpadkom obrazu a zvuku.



Obr. 23.: Simulácia poruchy linky pre prenos videa v reálnom čase



Obr. 24.: Riešenie poruchy linky za pomoci vytvorenia záložného spojenia (1)

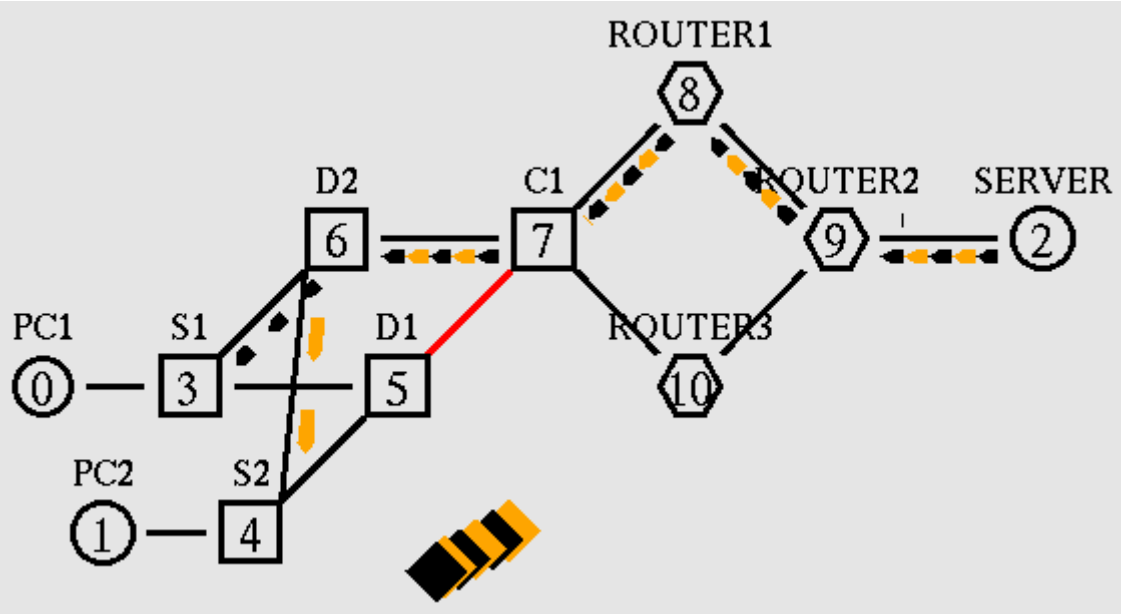
13.3.6.2 Simulácia poruchy linky medzi uzlami C1 a D1

Ďalšie poruchy linky môžu nastať v samotnej sieti na príjmacej strane.Topológia siete je navrhnutá tak, aby riešila výpadky aj medzi inými uzlami (tab. 4).

Tab. 4: Riešenie záložných spojení pri vzniknutí porúch v prijímacej sieti

nastanie poruchy medzi linkami	vytvorenie záložného spojenia
C1-D1	C1-D2
C1-D2	C1-D1
S1-D1	S1-D2
S1-D2	S1-D1
S2-D2	S2-D1
S2-D1	S2-D2

Pre simuláciu poruchy linky som zvolil výpadok medzi uzlami C1 a D1. Porucha je ošetrená aktivovaním záložného spojenia na linke medzi C1 a D2 (obr. 25).



Obr. 25.: Riešenie poruchy linky za pomoci vytvorenia záložného spojenia (2)

13.3.7 Simulácia prenosu videa v reálnom čase

Topológia siete, ktorá je použitá pre simulovanie je zobrazená na obr. 24-25. Jedná sa o multicast prenos, teda prenos pre určitú skupinu užívateľov. Audio a video dáta sú vysielané v smere od servera na počítače PC1 a PC2. Poruchy liniek sú ošetrené záložnými spojeniami ako je už spomenuté v predošlých kapitolách. Vďaka aktivovaniu smerovacieho protokolu sa pred prenosom vyhodnotia linky. Vyššiu prioritu má linka s väčšou prenosovou rýchlosťou.

Prenosové rýchlosti na linkách sú nasledovné:

Tab. 5: Veľkosti prenosových rýchlostí medzi uzlami

linka medzi uzlami		prenosová rýhlosť
PC1	S1	2 Mbps
PC2	S2	0,8 Mbps
S1	D1	5 Mbps
S1	D2	5 Mbps
S2	D1	5 Mbps
S2	D2	5 Mbps
D1	C1	8 Mbps
D2	C1	8 Mbps
C1	Router1	2 Mbps
C1	Router3	2 Mbps
Router1	Router2	2 Mbps
Router3	Router2	2 Mbps
Router2	Server	10 Mbps

Všetky zariadenia používajú najbežnejší typ fronty tzv. *DropTail*. Je to fronta typu FIFO. Video prenos (streaming) je založený na kodeku MPEG-1 so strednou kvalitou obrazu s využitím spomínaného multicastu. Cieľom simulácie bolo dosiahnutie prenosu dát bez stratovosti dát (paketov). Rýchlosť okolo 2 Mbps pre vysielané dáta sa zdá byť dostatočná ako na vysielacej, tak i prijímacej strane. A však pri rýchlosti okolo 1 Mbps dochádza

k zaplneniu linky a k následným, i keď spočiatku iba čiastočným, stratám dát. Odstránenie stratovosti je možné zvýšením prenosovej kapacity na linkách.

13.3.8 Definovanie zdrojového kódu pre vytvorenie simulácie

„Hlavička“ tohto programu a príkazy pre vytvorenie uzlov sú podobné ako v príklade pre vytvorenie simulácie prenosu hlasového toku.

Nastavenie pre multicast prenos:

```
$ns multicast
$ns mrtproto DM
```

Ďalej som vytvoril linky medzi uzlami, nastavil veľkosť fronty a určil ich grafické rozmiestnenie:

```
$ns duplex-link $s1 $pc1 2Mb 10ms DropTail

$ns queue-limit $s1 $pc1 10

$ns duplex-link-op $s1 $pc1 orient left
```

Sledovanie fronty na linkách:

```
$ns duplex-link-op $s2 $d1 queuePos 0.5
$ns duplex-link-op $s1 $d2 queuePos 0.5
```

Definícia UDP spojení a aplikácii s konštantnou bitovou rýchlosťou (CBR):

```
#Definícia UDP-1 spojenia
set udp [new Agent/UDP]
$ns attach-agent $server $udp
set null [new Agent/Null]
$ns attach-agent $pc1 $null
$ns connect $udp $null
$udp set fid_ 1 #Priradenie farby

#Definícia aplikácie s konštantnou bitovou rýchlosťou (CBR)
set cbr1 [new Application/Traffic/CBR]
$cbr1 attach-agent $udp
$cbr1 set type_ CBR
$cbr1 set packet_size_ 2416
$cbr1 set rate_ 1mb
$cbr1 set random_ false
```

Modelovanie výpadku linky:

```
$ns rtmodel-at 0.6  down $c1 $d1
```

Nastavenie časových udalostí:

```
$ns at 0.0 "$cbr1 start"
$ns at 0.1 "$cbr2 start"
$ns at 2.9 "$cbr2 stop"
$ns at 3.0 "$cbr1 stop"
```

Koncové príkazy sú obdobné ako v príklade pre vytvorenie simulácie prenosu hlasového toku. Príkaz, pre odpojenie generátora a konzumenta prevádzky od spojenia, nie je povinný.

Ako úplne posledný príkaz napíšeme „`$ns run`“, ktorým sa vyvolá zahájenie simulácie.

13.4 Problémy pri spúšťaní programov a ich riešenie

Pri spúšťaní programov sa vyskytol iba jeden problém. Bola to nasledujúce oznámenie s chybou:

warning: no class variable Session/RTP::debug_

Riešenie problému som našiel za pomoci internetu na stránke:

http://mailman.isi.edu/pipermail/ns-users/2006-September/057171.html

Autor „Pedro Vale Estrela“ rieši problém nasledovne:

a) Je treba otvoriť si cestu „ns / tcl / lib / ns-delfault.tcl“ a do ns-delfault.tcl dopísať:

Session/RTP::debug_ set 0

potom rekompilovať Network Simulator a po rekompilácii použiť v našom skripte:

Session/RTP::debug_ set 1

b) Ďalej je nutné rozšíriť náš skript v časti, kde je definovaný UDP port s použitím agentov.

13.5 Záver

Mojou úlohou bolo vytvorenie dvoch simulácii v programe Network Simulator. Prvou bola simuláciu prenosu hlasového toku medzi dvomi účastníkmi a druhou bol prenos videa v reálnom čase. Keďže sa jedná o aplikácie citlivé na stratu paketov a oneskorenie behom prenosu, je potrebné im zaistiť požadovanú kvalitu a to za pomoci QoS. Pre obidve simulácie bolo potrebné navrhnúť vhodné topológie IP sietí. Pre reálne chovanie simulácie som potreboval vedieť aké veľké objemy dát budú prenášané a aká je požadovaná minimálna veľkosť šírky pásma. Veľkosti paketov a veľkosť celkovej šírky pásma sú dané použitým kodekom, IP hlavičkou, prenosovým médiom, atď.

Zo zadania som mal určené, že pre prenos hlasu mám použiť kodek G.723. Tento kodek sa vyznačuje dlhšou periódou vysielaných paketov (až 30 ms, pričom u iných kodekov je to 20 ms a menej). Vzorkovacia perióda a oneskorenie (latency) majú priamo-úmerný vzťah. Čím dlhšia je vzorkovacia perióda, tým väčšie je oneskorenie, a to má neuspokojivý dopad na kvalitu hovoru. Preto pri simulácii bolo potrebné k celkovému oneskoreniu pričítať ešte 30 ms, čo bola doba kedy dochádzalo k prevodu hlasu podľa doporučenia G.723.

Pre prenos videa v reálnom čase som použil doporučenie podľa H.263. Jedná sa o video kodek, ktorý podporuje video kompresiu pre video-konferenciu a video-telefóniu s nízkou-bitovou kompresiou dát. Umožňuje efektívnejšie spracovanie malých paketov a paketov s nízkou bitovou rýchlosťou. V simulácii boli prenášané dáta o veľkosti 302 bajtov prenosovou rýchlosťou podľa štandardu MPEG-1, tj. minimálne 1,5 Mbps. Poruchy na linkách som riešil vytvorením záložných spojení, ktoré sa aktivovali hneď pri vzniknutí výpadku na linkách.

Záver

Vypracovaná bakalárska práca je zameraná na implementáciu kvality služieb (QoS) do konvergovanej siete, tj. siete prenášajúcej spoločne hlasovú a dátovú službu. Začiatok teoretickej časti je najprv venovaný vysvetleniu, čo to konvergovaná sieť je. V praxi je táto sieť známa pod komerčným označením ako NGN (Next Generation Networking) teda sieť budúcej generácie. Umožňuje prenášať hlas a video v digitalizovanej forme po dátovom médiu.

QoS je základná schopnosť siete dodržať prostredníctvom prepínačov a smerovačov v spolupráci s telefónnymi a koncovými video zariadeniami také parametry prenosu, aby bola zaistená kvalita hlasu a videa bez ohľadu na vyťažiteľnosť siete.

Ďalej je pozornosť venovaná dátovej sieti Ethernet, od jej počiatkov až po dnešok a adresovanie v nej za pomoci protokolu IPv6. Aby bolo možné hlas a video cez dátovú sieť vysielať, boli vytvorené kodeky, ktoré kódujú a dekódujú tieto dáta. Typickým príkladom môže byť hlasový kodek G.723. Keďže hlas je kritická aplikácia, ktorá vyžaduje maximálnu dostupnosť a spoľahlivosť siete, sú použité protokoly (napr. UDP, RTP), ktoré zaručujú rovnomernosť toku paketov v sieti.

Pre kódovanie a kompresiu zvuku, signalizáciu volania a zabezpečenie komunikácie medzi koncovými bodmi a riadiacou signalizáciou bol vytvorený súbor protokolov H.263. Ten ponúka riadiaci mechanizmus pre nadviazanie a ukončenie spojenia a uskutočňuje digitalizáciu a paketizáciu audio formátu pre prenos hlasu IP sieťami.

Konvergované siete sú siete budúcnosti. Umožňujú spojenie telekomunikačnej a dátovej siete dokopy. Vďaka tomu klesajú náklady na výstavbu a rozširovanie sieťovej infraštruktúry IT a organizačnej štruktúry správy telefónnej siete, tým sa zjednodušuje správa a údržba celého systému.

Literatúra:

[1] Toby J. Velte, Anthony T. Vlete: *Cisco® A Beginner's Guid-Fourt Edition*, Copyrigh © 2007 by The McGraw-Hill Companies.

[2] *CCNA Exploration curriculum,* -elektronické skriptá od Cisco®, 2007.

[3] Pužmanová R.: *Moderní komunikační sítě A-Z.* Computer Press, Brno 2007.

[4] Blunár K. a Diviš Z.: *Telekomunikačné siete*. ŽU, Žilina 2000.

[5] ALCONET, s.r.o.: *VoIP.* [on-line]. Dostupné z: <http://www.alconet.sk/voip>.

[6] Kolektív autorov: *Ďalšia generácia sietí.* [on-line]. Dostupné z: <http://advances.utc.sk/>.

[7] Rakovský J.: *Inteligentní síť pro IP komunikaci.* [on-line]. Dostupné z: <http://www.cisco.cz/>.

[8] Kolektív autorov: Rhys Haden: *Data Encoding Techniques*. [on-line]. Dostupné z: <http://www.rhyshaden.com/encoding.htm>.

[9] C Zhu, Intel Corp.: *RTP payload format for H.263 video stream.* [on-line] Dostupné z: <http://www.ietf.org/rfc/rfc2190.txt>.

[10] Skupina: USC Information Sciences Institute: *The Network Simulator –ns2-*. [on-line]. Dostupné z: <http://www.isi.edu/nsnam/ns/>.

[11] *CCNP curriculum,* -elektronické skriptá od Cisco®, 2005.

[12] Skupina: Newport networks: *VoIP bandwidth calculation.* [on-line] Dostupné z: <http://www.newport-networks.com/whitepapers/voip-bandwidth3.html>.

[13] Chen-Nee Chuan, University of California, Davis: *Four sources of packet delay*. [on-line] . Dostupné z: <http://www.ece.ucdavis.edu/~chuah/classes/EEC173A/lab/Note2_measurement.pdf >.

[14] Rakel Daniel: *Jitter buffer calculation*. [on-line] Dostupné z: <http://www.velocityreviews.com/forums/t233386-jitter-buffer-calculation.html>.

[15] Siggs Roger: *Streaming calculation*. [on-line] Dostupné z: <http://wiki.gc.maricopa.edu/display/docs/Streaming+Calculation>.

Prílohy

PRÍLOHA 1: Zdrojový kód pre simuláciu prenosu hlasového toku (1)

```
#Vytvorenie objektu
set ns [new Simulator]

#Aktivácia smerovacieho protokolu distance-vector
$ns rtproto DV

#Definícia farieb jednotlivých dátových tokov
$ns color 1 Black
$ns color 2 Orange

#Otvorenie súboru pre zapis v NAM
set nf [open out.nam w]
$ns namtrace-all $nf

#Definícia procedúry 'finish'
proc finish {} {
        global ns nf
        $ns flush-trace
        #zatvorenie NAM
        close $nf
        #spustenie NAM
        exec nam out.nam &
        exit 0
}

#Vytvorenie uzlov
set phone1 [$ns node]
$ns at 0.0 "$phone1 label IPphone1"
set phone2 [$ns node]
$ns at 0.0 "$phone2 label IPphone2"
set s1 [$ns node]
$ns at 0.0 "$s1 label S1"
$s1 shape "box"
set s2 [$ns node]
$ns at 0.0 "$s2 label S2"
$s2 shape "box"
set d1 [$ns node]
$ns at 0.0 "$d1 label D1"
$d1 shape "box"
set d2 [$ns node]
$ns at 0.0 "$d2 label D2"
$d2 shape "box"


#Vytvorenie liniek medzi uzlami
$ns duplex-link $s1 $phone1 1Mb 10ms DropTail
$ns duplex-link $s1 $d1 1Mb 20ms DropTail
$ns duplex-link $s1 $d2 1Mb 20ms DropTail
$ns duplex-link $d2 $s2 1Mb 20ms DropTail
$ns duplex-link $d1 $s2 1Mb 20ms DropTail
$ns duplex-link $s2 $phone2 1Mb 10ms DropTail

#Nastavenie veľkosti fronty liniek medzi uzlami
$ns queue-limit $s1 $phone1 10
$ns queue-limit $s1 $d1 10
$ns queue-limit $s1 $d2 10
```

```
$ns queue-limit $d2 $s2 10
$ns queue-limit $d1 $s2 10
$ns queue-limit $s2 $phone2 10

#Grafické rozmiestnenie uzlov
$ns duplex-link-op $s1 $phone1 orient left
$ns duplex-link-op $s1 $d1 orient right-down
$ns duplex-link-op $s1 $d2 orient right-up
$ns duplex-link-op $d2 $s2 orient right-down
$ns duplex-link-op $d1 $s2 orient right-up
$ns duplex-link-op $s2 $phone2 orient right

#Sledovanie stavu fronty na linkách
$ns duplex-link-op $phone1 $s1 queuePos 0.5

#Vytvorenie RTP1 agenta
set RTP1 [new Agent/RTP]
$ns attach-agent $phone2 $RTP1
$RTP1 set fid_ 1

#Vytvorenie RTCP1 agenta
set RTCP1 [new Agent/RTCP]
$ns attach-agent $phone1 $RTCP1
$RTCP1 set fid_ 2

#Spojenie RTP1 a RTCP1 a vytvorenie session
$ns connect $RTP1 $RTCP1
set session1 [new Session/RTP]
$RTP1 session $session1
$RTCP1 session $session1

#Vytvorenie aplikácie generujúcej konštantný dátový tok "hovor1"
set hovor1 [new Application/Traffic/CBR]
$hovor1 attach-agent $RTP1
$hovor1 set packetSize_ 976
$hovor1 set rate_ 6.4kb

#Definícia UDP-1 spojenia
set udp [new Agent/UDP]
$ns attach-agent $phone2 $udp
set null [new Agent/Null]
$ns attach-agent $phone1 $null
$ns connect $udp $null
$udp set fid_ 1 #Priradenie farby

#Definícia aplikácie s konštantnou bitovou rýchlosťou (CBR)
set cbr1 [new Application/Traffic/CBR]
$cbr1 attach-agent $udp
$cbr1 set type_ CBR
$cbr1 set packet_size_ 976
$cbr1 set rate_ 1mb
$cbr1 set random_ false

#Definícia UDP-2 spojenia
set udp [new Agent/UDP]
$ns attach-agent $phone1 $udp
set null [new Agent/Null]
$ns attach-agent $phone2 $null
$ns connect $udp $null
$udp set fid_ 2 #Priradenie farby
```

```
#Definícia aplikácie s konštantnou bitovou rýchlosťou (CBR)
set cbr2 [new Application/Traffic/CBR]
$cbr2 attach-agent $udp
$cbr2 set type_ CBR
$cbr2 set packet_size_ 976
$cbr2 set rate_ 1mb
$cbr2 set random_ false

#Modelovanie výpadku linky
$ns rtmodel-at 1.0 down $s1 $d1

#Časovanie udalostí pre generátory provozu
#$ns at 0.0 "$RTCP1 start"
#$ns at 0.0 "$hovor1 start"
#$ns at 0.0 "$hovor2 start"
$ns at 0.0 "$cbr1 start"
$ns at 0.2 "$cbr2 start"
#$ns at 2.8 "$hovor1 stop"
#$ns at 2.8 "$hovor2 stop"
$ns at 2.8 "$cbr2 stop"
$ns at 2.8 "$cbr1 stop"
#$ns at 3.0 "$RTCP1 stop"

#Odpojenie generátoru a konzumenta provozu od spojenia (neni povinné)
$ns at 3.0 "$ns detach-agent $phone2 $udp ; $ns detach-agent $phone1 $null"

#Zavolanie procedúry finish po 3 sekundách od spustenia simulácie
$ns at 3.0 "finish"

#Zápis údajov o veľkosti paketov a časového intervalu
#odosielania paketov na štandardný výstup
puts "CBR packet size = [$cbr1 set packet_size_]"
puts "CBR interval = [$cbr1 set interval_]"
puts "CBR packet size = [$cbr2 set packet_size_]"
puts "CBR interval = [$cbr2 set interval_]"

#Zahájenie simulácie
$ns run
```

PRÍLOHA 2: Zdrojový kód pre simuláciu prenosu hlasového toku (2)

```
#Vytvorenie objektu
set ns [new Simulator]

#Aktivácia smerovacieho protokolu distance-vector
$ns rtproto DV

#Definícia farieb jednotlivých dátových tokov
$ns color 1 Black
$ns color 2 Orange

#Otvorenie súboru pre zapis v NAM
set nf [open out.nam w]
$ns namtrace-all $nf

#Definícia procedúry 'finish'
proc finish {} {
        global ns nf
        $ns flush-trace
        #zatvorenie NAM
        close $nf
        #spustenie NAM
        exec nam out.nam &
        exit 0
}

#Vytvorenie uzlov
set phone1 [$ns node]
$ns at 0.0 "$phone1 label IPphone1"
set phone2 [$ns node]
$ns at 0.0 "$phone2 label IPphone2"
set s1 [$ns node]
$ns at 0.0 "$s1 label S1"
$s1 shape "box"
set s2 [$ns node]
$ns at 0.0 "$s2 label S2"
$s2 shape "box"
set d1 [$ns node]
$ns at 0.0 "$d1 label D1"
$d1 shape "box"
set d2 [$ns node]
$ns at 0.0 "$d2 label D2"
$d2 shape "box"

#Vytvorenie liniek medzi uzlami
$ns duplex-link $s1 $phone1 2Mb 10ms DropTail
$ns duplex-link $s1 $d1 2Mb 20ms DropTail
$ns duplex-link $s1 $d2 2Mb 20ms DropTail
$ns duplex-link $d2 $s2 2Mb 20ms DropTail
$ns duplex-link $d1 $s2 2Mb 20ms DropTail
$ns duplex-link $s2 $phone2 2Mb 10ms DropTail

$ns rtproto Manual

#Nastavenie veľkosti fronty liniek medzi uzlami
$ns queue-limit $s1 $phone1 10
$ns queue-limit $s1 $d1 10
$ns queue-limit $s1 $d2 10
$ns queue-limit $d2 $s2 10
$ns queue-limit $d1 $s2 10
```

```
$ns queue-limit $s2 $phone2 10

#Grafické rozmiestnenie uzlov
$ns duplex-link-op $s1 $phone1 orient left
$ns duplex-link-op $s1 $d1 orient right-down
$ns duplex-link-op $s1 $d2 orient right-up
$ns duplex-link-op $d2 $s2 orient right-down
$ns duplex-link-op $d1 $s2 orient right-up
$ns duplex-link-op $s2 $phone2 orient right

#Sledovanie stavu fronty na linkách
$ns duplex-link-op $phone1 $s1 queuePos 0.5

#Definícia RTP-1 spojenia
set rtp [new Agent/RTP]
$ns attach-agent $phone1 $rtp
set sink [new Agent/RTP]
$ns attach-agent $phone2 $sink
$ns connect $rtp $sink
$rtp set fid_ 1 #Priradenie farby
$rtp set packetSize_ 976

#Definícia RTP-2 spojenia
set rtp1 [new Agent/RTP]
$ns attach-agent $phone2 $rtp1
set sink [new Agent/RTP]
$ns attach-agent $phone1 $sink
$ns connect $rtp1 $sink
$rtp1 set fid_ 2 #Priradenie farby
$rtp1 set packetSize_ 976

#Nastavenie Exponential provozu cez RTP spojenie
set voice [new Application/Traffic/Exponential]
$voice set brust_time_ 1.004s
$voice set idle_time_ 1.587s
$voice set rate_ 80k

#Modelovanie výpadku linky
#$ns rtmodel-at 1.0 down $s1 $d1

#Časovanie udalostí pre generátory provozu
$ns at 0.1 "$rtp start"
$ns at 0.1 "$rtp1 start"
$ns at 2.9 "$rtp1 stop"
$ns at 2.9 "$rtp stop"

#Zavolanie procedúry finish po 3 sekundách od spustenia simulácie
$ns at 3.0 "finish"

#Zápis údajov o veľkosti paketov a časového intervalu
#odosielania paketov na štandardný výstup
puts "RTP G.723 packet size = [$rtp set packetSize_]"
puts "RTP G.723 interval = [$rtp set interval_]"
#puts "RTP G.723 packet size = [$rtp2 set packetSize_]"
#puts "RTP G.723 interval = [$rtp2 set interval_]"

#Zahájenie simulácie
$ns run
```

PRÍLOHA 3: Zdrojový kód pre simuláciu prenosu videa v reálnom čase (1)

```
#Vytvorenie objektu
set ns [new Simulator]

#Aktivácia smerovacieho protokolu distance-vector
$ns rtproto DV
$ns multicast
$ns mrtproto DM

#Definícia farieb jednotlivých dátových tokov
$ns color 1 Black
$ns color 2 Orange

#Otvorenie súboru pre zapis v NAM
set nf [open out.nam w]
$ns namtrace-all $nf

#Definícia procedúry 'finish'
proc finish {} {
        global ns nf
        $ns flush-trace
        #zatvorenie NAM
        close $nf
        #spustenie NAM
        exec nam out.nam &
        exit 0
}

#Vytvorenie uzlov
set pc1 [$ns node]
$ns at 0.0 "$pc1 label PC1"
set pc2 [$ns node]
$ns at 0.0 "$pc2 label PC2"
set server [$ns node]
$ns at 0.0 "$server label SERVER"
set s1 [$ns node]
$ns at 0.0 "$s1 label S1"
$s1 shape "box"
set s2 [$ns node]
$ns at 0.0 "$s2 label S2"
$s2 shape "box"
set d1 [$ns node]
$ns at 0.0 "$d1 label D1"
$d1 shape "box"
set d2 [$ns node]
$ns at 0.0 "$d2 label D2"
$d2 shape "box"
set c1 [$ns node]
$ns at 0.0 "$c1 label C1"
$c1 shape "box"
set router1 [$ns node]
$ns at 0.0 "$router1 label ROUTER1"
$router1 shape "hexagon"
set router2 [$ns node]
$ns at 0.0 "$router2 label ROUTER2"
$router2 shape "hexagon"

#Vytvorenie liniek medzi uzlami
$ns duplex-link $s1 $pc1 2Mb 10ms DropTail
$ns duplex-link $s1 $d1 2Mb 20ms DropTail
```

```
$ns duplex-link $s1 $d2 2Mb 20ms DropTail
$ns duplex-link $s2 $pc2 2Mb 10ms DropTail
$ns duplex-link $s2 $d2 2Mb 20ms DropTail
$ns duplex-link $s2 $d1 2Mb 20ms DropTail
$ns duplex-link $d1 $c1 2Mb 20ms DropTail
$ns duplex-link $d2 $c1 2Mb 20ms DropTail
$ns duplex-link $c1 $router1 2Mb 20ms DropTail
$ns duplex-link $router1 $router2 2Mb 20ms DropTail
$ns duplex-link $router2 $server 2Mb 20ms DropTail

#Nastavenie veľkosti fronty liniek medzi uzlami
$ns queue-limit $s1 $pc1 10
$ns queue-limit $s1 $d1 10
$ns queue-limit $s1 $d2 10
$ns queue-limit $s2 $pc2 10
$ns queue-limit $s2 $d2 10
$ns queue-limit $s2 $d1 10
$ns queue-limit $d1 $c1 10
$ns queue-limit $d2 $c1 10
$ns queue-limit $c1 $router1 10
$ns queue-limit $router1 $router2 10
$ns queue-limit $router2 $server 10

#Grafické rozmiestnenie uzlov
$ns duplex-link-op $s1 $pc1 orient left
$ns duplex-link-op $s1 $d1 orient right
$ns duplex-link-op $s1 $d2 orient right-up
$ns duplex-link-op $s2 $pc2 orient left
$ns duplex-link-op $s2 $d2 orient right-up
$ns duplex-link-op $s2 $d1 orient right-up
$ns duplex-link-op $d1 $c1 orient right-up
$ns duplex-link-op $d2 $c1 orient right
$ns duplex-link-op $c1 $router1 orient right
$ns duplex-link-op $router1 $router2 orient right
$ns duplex-link-op $router2 $server orient right

#Sledovanie stavu fronty na linkách
$ns duplex-link-op $s2 $d1 queuePos 0.5
$ns duplex-link-op $s1 $d2 queuePos 0.5

#Definícia UDP-1 spojenia
set udp [new Agent/UDP]
$ns attach-agent $server $udp
set null [new Agent/Null]
$ns attach-agent $pc1 $null
$ns connect $udp $null
$udp set fid_ 1 #Priradenie farby

#Definícia aplikácie s konštantnou bitovou rýchlosťou (CBR)
set cbr1 [new Application/Traffic/CBR]
$cbr1 attach-agent $udp
$cbr1 set type_ CBR
$cbr1 set packet_size_ 1000
$cbr1 set rate_ 1mb
$cbr1 set random_ false

#Definícia UDP-2 spojenia
set udp [new Agent/UDP]
$ns attach-agent $server $udp
set null [new Agent/Null]
$ns attach-agent $pc2 $null
```

```
$ns connect $udp $null
$udp set fid_ 2 #Priradenie farby

#Definícia aplikácie s konštantnou bitovou rýchlosťou (CBR)
set cbr2 [new Application/Traffic/CBR]
$cbr2 attach-agent $udp
$cbr2 set type_ CBR
$cbr2 set packet_size_ 1000
$cbr2 set rate_ 1mb
$cbr2 set random_ false

#Modelovanie výpadku linky
$ns rtmodel-at 0.6 down $router1 $router2

#Časovanie udalostí pre generátory provozu
$ns at 0.0 "$cbr1 start"
$ns at 0.1 "$cbr2 start"
$ns at 2.9 "$cbr2 stop"
$ns at 3.0 "$cbr1 stop"

#Odpojenie generátoru a konzumenta provozu od spojenia (neni povinné)
$ns at 3.0 "$ns detach-agent $server $udp ; $ns detach-agent $pc1 $null"

#Zavolanie procedúry finish po 3 sekundách od spustenia simulácie
$ns at 3.0 "finish"

#Zápis údajov o veľkosti paketov a časového intervalu
#odosielania paketov na štandardný výstup
puts "CBR packet size = [$cbr1 set packet_size_]"
puts "CBR interval = [$cbr1 set interval_]"
puts "CBR packet size = [$cbr2 set packet_size_]"
puts "CBR interval = [$cbr2 set interval_]"

#Zahájenie simulácie
$ns run
```

PRÍLOHA 4: Zdrojový kód pre simuláciu prenosu videa v reálnom čase (2)

```
#Vytvorenie objektu
set ns [new Simulator]

#Aktivácia smerovacieho protokolu distance-vector
$ns rtproto DV
$ns multicast
$ns mrtproto DM

#Definícia farieb jednotlivých dátových tokov
$ns color 1 Black
$ns color 2 Orange

#Otvorenie súboru pre zapis v NAM
set nf [open out.nam w]
$ns namtrace-all $nf

#Definícia procedúry 'finish'
proc finish {} {
        global ns nf
        $ns flush-trace
        #zatvorenie NAM
        close $nf
        #spustenie NAM
        exec nam out.nam &
        exit 0
}

#Vytvorenie uzlov
set pc1 [$ns node]
$ns at 0.0 "$pc1 label PC1"
set pc2 [$ns node]
$ns at 0.0 "$pc2 label PC2"
set server [$ns node]
$ns at 0.0 "$server label SERVER"
set s1 [$ns node]
$ns at 0.0 "$s1 label S1"
$s1 shape "box"
set s2 [$ns node]
$ns at 0.0 "$s2 label S2"
$s2 shape "box"
set d1 [$ns node]
$ns at 0.0 "$d1 label D1"
$d1 shape "box"
set d2 [$ns node]
$ns at 0.0 "$d2 label D2"
$d2 shape "box"
set c1 [$ns node]
$ns at 0.0 "$c1 label C1"
$c1 shape "box"
set router1 [$ns node]
$ns at 0.0 "$router1 label ROUTER1"
$router1 shape "hexagon"
set router2 [$ns node]
$ns at 0.0 "$router2 label ROUTER2"
$router2 shape "hexagon"
set router3 [$ns node]
$ns at 0.0 "$router3 label ROUTER3"
$router3 shape "hexagon"
```

```
#Vytvorenie liniek medzi uzlami
$ns duplex-link $s1 $pc1 2Mb 10ms DropTail
$ns duplex-link $s1 $d1 2Mb 20ms DropTail
$ns duplex-link $s1 $d2 2Mb 20ms DropTail
$ns duplex-link $s2 $pc2 2Mb 10ms DropTail
$ns duplex-link $s2 $d2 2Mb 20ms DropTail
$ns duplex-link $s2 $d1 2Mb 20ms DropTail
$ns duplex-link $d1 $c1 2Mb 20ms DropTail
$ns duplex-link $d2 $c1 2Mb 20ms DropTail
$ns duplex-link $c1 $router1 2Mb 20ms DropTail
$ns duplex-link $c1 $router3 2Mb 20ms DropTail
$ns duplex-link $router1 $router2 2Mb 20ms DropTail
$ns duplex-link $router3 $router2 2Mb 20ms DropTail
$ns duplex-link $router2 $server 2Mb 20ms DropTail

#Nastavenie veľkosti fronty liniek medzi uzlami
$ns queue-limit $s1 $pc1 10
$ns queue-limit $s1 $d1 10
$ns queue-limit $s1 $d2 10
$ns queue-limit $s2 $pc2 10
$ns queue-limit $s2 $d2 10
$ns queue-limit $s2 $d1 10
$ns queue-limit $d1 $c1 10
$ns queue-limit $d2 $c1 10
$ns queue-limit $c1 $router1 10
$ns queue-limit $c1 $router3 10
$ns queue-limit $router1 $router2 10
$ns queue-limit $router3 $router2 10
$ns queue-limit $router2 $server 10

#Grafické rozmiestnenie uzlov
$ns duplex-link-op $s1 $pc1 orient left
$ns duplex-link-op $s1 $d1 orient right
$ns duplex-link-op $s1 $d2 orient right-up
$ns duplex-link-op $s2 $pc2 orient left
$ns duplex-link-op $s2 $d2 orient right-up
$ns duplex-link-op $s2 $d1 orient right-up
$ns duplex-link-op $d1 $c1 orient right-up
$ns duplex-link-op $d2 $c1 orient right
$ns duplex-link-op $c1 $router1 orient right-up
$ns duplex-link-op $c1 $router3 orient right-down
$ns duplex-link-op $router1 $router2 orient right-down
$ns duplex-link-op $router3 $router2 orient right-up
$ns duplex-link-op $router2 $server orient right

#Sledovanie stavu fronty na linkách
$ns duplex-link-op $s2 $d1 queuePos 0.5
$ns duplex-link-op $s1 $d2 queuePos 0.5

#Definícia UDP-1 spojenia
set udp [new Agent/UDP]
$ns attach-agent $server $udp
set null [new Agent/Null]
$ns attach-agent $pc1 $null
$ns connect $udp $null
$udp set fid_ 1 #Priradenie farby

#Definícia aplikácie s konštantnou bitovou rýchlosťou (CBR)
set cbr1 [new Application/Traffic/CBR]
$cbr1 attach-agent $udp
$cbr1 set type_ CBR
```

```
$cbr1 set packet_size_ 2416
$cbr1 set rate_ 1mb
$cbr1 set random_ false

#Definícia UDP-2 spojenia
set udp [new Agent/UDP]
$ns attach-agent $server $udp
set null [new Agent/Null]
$ns attach-agent $pc2 $null
$ns connect $udp $null
$udp set fid_ 2 #Priradenie farby

#Definícia aplikácie s konštantnou bitovou rýchlosťou (CBR)
set cbr2 [new Application/Traffic/CBR]
$cbr2 attach-agent $udp
$cbr2 set type_ CBR
$cbr2 set packet_size_ 2416
$cbr2 set rate_ 1mb
$cbr2 set random_ false

#Modelovanie výpadku linky
$ns rtmodel-at 0.6 down $router1 $router2
$ns rtmodel-at 1.5 down $c1 $d1

#Časovanie udalostí pre generátory provozu
$ns at 0.0 "$cbr1 start"
$ns at 0.1 "$cbr2 start"
$ns at 2.9 "$cbr2 stop"
$ns at 3.0 "$cbr1 stop"

#Odpojenie generátoru a konzumenta provozu od spojenia (neni povinné)
$ns at 3.0 "$ns detach-agent $server $udp ; $ns detach-agent $pc1 $null"

#Zavolanie procedúry finish po 3 sekundách od spustenia simulácie
$ns at 3.0 "finish"

#Zápis údajov o veľkosti paketov a časového intervalu
#odosielania paketov na štandardný výstup
puts "CBR packet size = [$cbr1 set packet_size_]"
puts "CBR interval = [$cbr1 set interval_]"
puts "CBR interval = [$cbr1 set rate_]"
puts "CBR packet size = [$cbr2 set packet_size_]"
puts "CBR interval = [$cbr2 set interval_]"
puts "CBR interval = [$cbr2 set rate_]"

#Zahájenie simulácie
$ns run
```